



Machine Vision Engineering Handbook

Julio Zanon Diaz

Contributor n2

Published 2026-08-01

Free for eligible public academic use under the MVA-OAL License.
Commercial use is prohibited. For complete licensing terms, updates and
additional content, visit <https://machinevisionacademy.org>.

Table of contents

1. Machine Vision Engineering Handbook	26
Welcome	27
Who is this handbook for?	27
Revision history	28
Acknowledgements	29
Notation	30
 I. Foundations	 31
2. Vision engineering: scope, definitions and history	33
2.1. Machine vision, computer vision and image processing . . .	33
2.1.1. Working definitions and industrial success criteria .	33
2.1.2. The four fields: image processing, computer vision, machine vision and AI	35
2.1.3. Relations, overlaps and typical confusions	38
2.2. Historical development: several timelines, one discipline . .	39
2.2.1. The innovation timeline: when images became com- putable	40
2.2.2. The industrial timeline: from bespoke lab prototypes to deployable vision	48
2.2.3. The product timeline: how the industrial toolbox formed	50

2.2.4.	The everyday consumer timeline: when vision es-	
	caped the laboratory and factory	70
2.2.5.	What the history reveals	74
2.3.	Industrial application families	75
2.3.1.	Measurement and non-contact gauging	78
2.3.2.	Defect inspection	79
2.3.3.	Identification of unique parts: codes, characters and	
	traceability	82
2.3.4.	Guidance and robot vision	84
2.4.	System architectures	86
2.4.1.	Device Edge Computing	88
2.4.2.	Machine Edge Computing	94
2.4.3.	On-Premises Vision Server	99
2.4.4.	Cloud Inference	103
2.4.5.	Comparison and selection criteria	106
2.5.	The vision system as an engineered chain	113
2.5.1.	The seven stages: presentation, illumination, optics,	
	sensor, processing, decision and action	113
2.5.2.	Constraint-first engineering: solve the image before	
	the algorithm	120
2.5.3.	How the chain interacts: propagate the requirement	
	end to end	123
2.5.4.	Professional roles, standards and certification landscape	125
2.6.	Approaching a vision problem: a design framework	129
2.6.1.	From inspection requirement to imaging specification	129
2.6.2.	Characterising the feature: contrast, scale, geometry,	
	tolerance	129
2.6.3.	Detection versus measurement: how the task drives	
	every later choice	130
2.6.4.	A worked decision path, cross-referenced to later	
	chapters	130

3. Light, colour and radiometry	131
3.1. Physical foundations of light	131
3.1.1. Electromagnetic spectrum and the visible band . . .	131
3.1.2. Radiometric quantities: radiant flux, irradiance, radiance	131
3.1.3. Photometric quantities and the photopic response .	132
3.1.4. The inverse-square law and cosine law	132
3.2. Interaction of light with matter	132
3.2.1. Specular and diffuse reflection; the BRDF	132
3.2.2. Transmission, refraction and Snell's law	132
3.2.3. Absorption and fluorescence	133
3.2.4. Scattering and translucency	133
3.2.5. Implications for surface inspection	133
3.3. Colour	133
3.3.1. Trichromacy and the human visual system	133
3.3.2. Colour matching and the CIE XYZ system	134
3.3.3. Working colour spaces: RGB, sRGB, HSV, HSL, Lab	134
3.3.4. Colour temperature and illuminants	134
3.3.5. White balance and colour constancy	134
3.4. Polarisation	135
3.4.1. States of polarisation and Stokes parameters	135
3.4.2. Polarisers, retarders and the Fresnel equations . . .	135
3.4.3. Polarisation-resolving sensors	135
3.4.4. Industrial uses: glare suppression, stress imaging, transparent-material inspection	135
4. Optics and lens systems	136
4.1. Geometric optics	136
4.1.1. The pinhole model	136
4.1.2. Thin- and thick-lens models	136
4.1.3. Focal length, f-number, aperture	137
4.1.4. Field of view and working distance	137
4.1.5. Depth of field and hyperfocal distance	137

4.2.	Real lens behaviour	137
4.2.1.	Spherical and chromatic aberration	137
4.2.2.	Geometric distortion: radial and tangential	138
4.2.3.	Vignetting and cos fall-off	138
4.2.4.	Modulation transfer function and resolving power	138
4.2.5.	Diffraction limit	138
4.3.	Specialty optics	139
4.3.1.	Telecentric lenses for metrology	139
4.3.2.	Macro and micro imaging	139
4.3.3.	Line-scan and Scheimpflug configurations	139
4.3.4.	Liquid and variable-focus lenses	139
4.4.	Optical filters	140
4.4.1.	Bandpass, longpass and shortpass filters	140
4.4.2.	Polarising filters in imaging	140
4.4.3.	Neutral density, UV cut and IR cut filters	140
4.4.4.	Interference filters and dichroic elements	140
4.4.5.	Filter placement and stacking	141
4.5.	Selecting an optical configuration	141
4.5.1.	Matching focal length to sensor size, FOV and working distance	141
4.5.2.	Resolution and DOF budgeting from the feature specification	141
4.5.3.	When telecentricity, filtering or specialty optics are required	141
4.5.4.	Mount standards and mechanical envelope: C, CS, F, M42, S-mount	142
4.5.5.	A lens-selection workflow, cross-referenced to Chapters 2 and 5	142

II. Image Formation	143
5. Illumination for vision	145
5.1. Illumination principles	145
5.1.1. Light and geometry as the first solution	145
5.1.2. Contrast, feature enhancement and defect visibility	145
5.1.3. The lighting-first design method	146
5.2. Illumination geometries	146
5.2.1. Frontal and diffuse illumination	146
5.2.2. Directional and grazing illumination	146
5.2.3. Backlighting for silhouettes and dimensioning	146
5.2.4. Coaxial (on-axis) illumination	147
5.2.5. Dome and cloudy-day illumination	147
5.2.6. Dark-field illumination	147
5.2.7. Structured light	147
5.3. Light sources	148
5.3.1. LEDs: colour, drive, lifetime, thermal management	148
5.3.2. Fluorescent, halogen and xenon sources	148
5.3.3. Laser sources and safety classes	148
5.3.4. Continuous, strobed and pulsed operation	148
5.4. Wavelength selection	149
5.4.1. Visible-band choices and colour separation	149
5.4.2. Near-infrared imaging	149
5.4.3. Ultraviolet imaging	149
5.4.4. Short-wave infrared and thermal imaging	149
5.4.5. Multispectral and hyperspectral approaches	150
5.5. Environmental integration	150
5.5.1. Ambient light rejection	150
5.5.2. Enclosures, shrouds and shielding	150
5.5.3. Vibration, contamination and thermal drift	150
5.6. Designing an illumination solution	151
5.6.1. Characterising the feature and the surface	151
5.6.2. Single-plane versus multi-plane and 3D features	151

5.6.3.	Edge visibility for measurement versus contrast for detection	151
5.6.4.	Matching geometry to task: a decision guide	151
5.6.5.	Combining geometry, wavelength and filtering	152
5.6.6.	Worked examples: from part to lighting choice	152
6.	Image sensors and cameras	153
6.1.	Sensor technologies	153
6.1.1.	CCD architecture and readout	153
6.1.2.	CMOS architecture: global vs rolling shutter	153
6.1.3.	Line-scan and TDI sensors	154
6.1.4.	SPAD, event-based and neuromorphic sensors	154
6.1.5.	Multispectral and hyperspectral sensor architectures	154
6.2.	Sensor performance and noise	154
6.2.1.	Quantum efficiency and spectral response	154
6.2.2.	Read noise, shot noise, dark current, fixed-pattern noise	155
6.2.3.	Dynamic range and bit depth	155
6.2.4.	Full-well capacity and saturation	155
6.2.5.	Signal-to-noise budgeting	155
6.3.	Sampling, resolution and colour acquisition	156
6.3.1.	Pixel pitch, sensor size and effective resolution	156
6.3.2.	Bayer patterns and demosaicing	156
6.3.3.	3-CCD and multi-sensor colour	156
6.3.4.	Monochrome versus colour trade-offs	156
6.4.	Camera interfaces and image transport	157
6.4.1.	Interface selection criteria: bandwidth, cable length, CPU load, cost	157
6.4.2.	USB3 Vision: bandwidth, cable limits and use cases	157
6.4.3.	Camera Link, CoaXPress and frame-grabber-based interfaces	157
6.4.4.	Legacy and niche interfaces: FireWire, analogue, MIPI CSI	157
6.4.5.	Triggering, exposure control and time stamping	158

6.5.	GigE Vision in depth	158
6.5.1.	Generational bandwidth: 1, 5, 10 and 25 GbE and practical reach	158
6.5.2.	The GigE Vision protocol: discovery, stream and control channels	158
6.5.3.	Network configuration: jumbo packets, dedicated NICs, receive buffers	158
6.5.4.	Multi-camera topologies, switches and bandwidth budgeting	159
6.5.5.	Power over Ethernet and cabling	159
6.5.6.	GenICam and cross-vendor interoperability	159
6.6.	Selecting a camera and interface	159
6.6.1.	From resolution and speed requirement to sensor choice	159
6.6.2.	Matching interface to bandwidth, distance and architecture	160
6.6.3.	Area-scan versus line-scan decision	160
6.6.4.	A camera-and-interface selection workflow	160

III. Image Processing 161

7. Digital images: representation and pixel-level processing 163

7.1.	The digital image as a signal	163
7.1.1.	Sampling, quantisation and the resulting data model	163
7.1.2.	Coordinate systems, indexing and boundary conventions	163
7.1.3.	Colour, greyscale and multi-channel representations	164
7.1.4.	File formats and lossless vs lossy compression	164
7.2.	Point operations	164
7.2.1.	Brightness, contrast and gain	164
7.2.2.	Look-up tables, gamma and tone curves	164
7.2.3.	Histogram equalisation and adaptive stretching	165
7.2.4.	Thresholding as a point operation	165

7.3.	Arithmetic and logical operations	165
7.3.1.	Pixel-wise arithmetic and blending	165
7.3.2.	Absolute and signed differences	165
7.3.3.	Masking and region-of-interest operations	166
7.3.4.	Bitwise operations on binary images	166
7.4.	Noise models and denoising choices	166
7.4.1.	Gaussian, Poisson and impulse noise models	166
7.4.2.	Estimating noise from image statistics	166
7.4.3.	Choosing a denoiser: signal preservation vs artefact suppression	167
8.	Neighbourhood operations and image filtering	168
8.1.	Convolution and correlation	168
8.1.1.	Discrete convolution in two dimensions	168
8.1.2.	Separable and rank-one kernels	168
8.1.3.	Boundary handling: zero, reflect, replicate, wrap	169
8.1.4.	Computational cost and efficient implementations	169
8.2.	Smoothing filters	169
8.2.1.	Box (mean) filter and integral images	169
8.2.2.	Gaussian filter and selection	169
8.2.3.	Median and rank-order filters	170
8.2.4.	Bilateral and edge-preserving filters	170
8.2.5.	Non-local means	170
8.3.	Derivative and edge filters	170
8.3.1.	First-order operators: Sobel, Prewitt, Scharr	170
8.3.2.	Second-order operators: Laplacian and Laplacian of Gaussian	171
8.3.3.	Gradient magnitude and orientation	171
8.3.4.	Sub-pixel edge localisation	171
8.4.	The frequency domain	171
8.4.1.	The 2D discrete Fourier transform	171
8.4.2.	Magnitude, phase and their perceptual roles	172
8.4.3.	The convolution theorem	172
8.4.4.	Low-pass, high-pass and band-pass filtering	172

8.4.5.	Frequency-domain inspection of periodic structure	172
8.4.6.	The sampling theorem and aliasing	173
8.5.	Scale space and image pyramids	173
8.5.1.	Scale-space theory in outline	173
8.5.2.	The Gaussian pyramid	173
8.5.3.	The Laplacian pyramid	173
8.5.4.	Multi-resolution processing patterns	174
8.6.	Choosing a filtering strategy	174
8.6.1.	Matching a filter to the noise model and the feature	174
8.6.2.	Preserving edges while suppressing noise	174
8.6.3.	Spatial versus frequency-domain: when each wins	174
8.6.4.	Ordering operations in a preprocessing pipeline	175
9.	Segmentation, morphology and blob analysis	176
9.1.	Thresholding methods	176
9.1.1.	Global thresholds and manual selection	176
9.1.2.	Otsu's method	176
9.1.3.	Adaptive and locally-adaptive thresholding	177
9.1.4.	Multi-level and hysteresis thresholding	177
9.2.	Region-based segmentation	177
9.2.1.	Region growing and merging	177
9.2.2.	Watershed segmentation	177
9.2.3.	Superpixels: SLIC and successors	178
9.2.4.	Graph-cut segmentation	178
9.3.	Edge-based segmentation	178
9.3.1.	The Canny detector	178
9.3.2.	Edge linking and boundary tracing	178
9.3.3.	Active contours (snakes) and level sets	179
9.4.	Mathematical morphology	179
9.4.1.	Structuring elements	179
9.4.2.	Erosion and dilation	179
9.4.3.	Opening and closing	179
9.4.4.	Hit-or-miss transform and skeletonisation	180
9.4.5.	Distance transforms and reconstruction	180

9.4.6.	Greyscale morphology	180
9.5.	Connected-components and blob analysis	180
9.5.1.	Connectivity and labelling algorithms	180
9.5.2.	Region properties: area, perimeter, centroid	181
9.5.3.	Geometric moments and Hu invariants	181
9.5.4.	Shape descriptors: circularity, eccentricity, compact- ness	181
9.5.5.	Filtering blobs by property	181
9.6.	Choosing a segmentation approach	182
9.6.1.	Thresholding versus region versus edge versus learned	182
9.6.2.	Reliability under illumination and contrast variation	182
9.6.3.	Combining segmentation with morphological clean-up	182
10.	Feature detection, description and matching	183
10.1.	Interest-point detection	183
10.1.1.	The Moravec detector and the corner concept	183
10.1.2.	Harris and Shi-Tomasi corners	183
10.1.3.	Scale-invariant detection: difference of Gaussians	184
10.1.4.	Hessian-based detectors	184
10.1.5.	FAST and AGAST corners for real-time use	184
10.2.	Feature description	184
10.2.1.	SIFT descriptor	184
10.2.2.	SURF descriptor	185
10.2.3.	Binary descriptors: BRIEF, ORB, BRISK, FREAK	185
10.2.4.	Learned descriptors: SuperPoint and successors	185
10.3.	Feature matching	185
10.3.1.	Distance metrics: L2, Hamming	185
10.3.2.	Brute-force and FLANN matching	186
10.3.3.	The Lowe ratio test and cross-check	186
10.3.4.	Robust matching with RANSAC and MSAC	186
10.4.	Template and pattern matching	186
10.4.1.	Sum of squared and absolute differences	186
10.4.2.	Normalised cross-correlation	187
10.4.3.	Rotation- and scale-invariant template matching	187

10.4.4. Geometric pattern matching (edge-model methods)	187
10.5. Line, circle and shape detection	187
10.5.1. The Hough transform for lines	187
10.5.2. The Hough transform for circles	188
10.5.3. The generalised Hough transform	188
10.5.4. Direct fitting: least-squares lines, circles, ellipses	188
10.5.5. Robust fitting with RANSAC	188
10.6. Choosing a feature method	189
10.6.1. Repeatability, invariance and speed trade-offs	189
10.6.2. Hand-crafted versus learned features	189
10.6.3. Matching the detector–descriptor pair to the task	189
11. Geometric transformations, calibration and registration	190
11.1. Coordinate systems and homogeneous coordinates	190
11.1.1. Image, camera, world and object frames	190
11.1.2. Homogeneous representation of points and lines	190
11.1.3. Matrix representation of transformations	191
11.2. The 2D transformation hierarchy	191
11.2.1. Rigid, similarity and affine transformations	191
11.2.2. Projective transformations (homographies)	191
11.2.3. Estimation from point correspondences	191
11.2.4. Robust estimation of homographies	192
11.3. Image warping and interpolation	192
11.3.1. Forward vs inverse mapping	192
11.3.2. Nearest-neighbour, bilinear and bicubic interpolation	192
11.3.3. Anti-aliased warping	192
11.4. The pinhole camera model	193
11.4.1. Intrinsic parameters and the camera matrix	193
11.4.2. Extrinsic parameters	193
11.4.3. Radial and tangential distortion models	193
11.5. Camera calibration	193
11.5.1. Zhang’s method and the planar-target approach	193
11.5.2. Calibration targets and image-acquisition strategy	194
11.5.3. Reprojection error and calibration diagnostics	194

11.5.4. Uncertainty and re-calibration policy	194
11.5.5. Hand-eye calibration for robot vision	194
11.6. Image registration	195
11.6.1. Feature-based registration pipelines	195
11.6.2. Intensity-based (direct) registration	195
11.6.3. Registration for inspection: golden templates	195
11.6.4. Mosaicing and stitching	195
11.7. Image-comparison metrics	196
11.7.1. Absolute and squared differences; PSNR	196
11.7.2. SSIM and multi-scale SSIM	196
11.7.3. Histogram-distance metrics	196
11.7.4. Learned perceptual metrics (outline)	196

IV. 3D Vision and Motion 197

12. Stereo and 3D vision 199

12.1. Two-view geometry	199
12.1.1. The epipolar constraint	199
12.1.2. The essential and fundamental matrices	199
12.1.3. Estimation from correspondences	200
12.1.4. Stereo rectification	200
12.2. Stereo matching	200
12.2.1. Block-matching costs: SAD, SSD, NCC, census	200
12.2.2. Aggregation and disparity selection	200
12.2.3. Semi-global matching	201
12.2.4. Learned stereo matchers (outline)	201
12.2.5. Disparity to depth and triangulation	201
12.3. Uncertainty in 3D measurement	201
12.3.1. Depth-error propagation from disparity	201
12.3.2. Task uncertainty versus datasheet accuracy	202
12.3.3. Uncertainty budgets for 3D measurement	202
12.4. 3D imaging modalities	202
12.4.1. Passive stereo	202

12.4.2. Structured-light triangulation and fringe projection .	202
12.4.3. Time-of-flight cameras	203
12.4.4. LiDAR and long-range sensing	203
12.4.5. Photogrammetry, structure from motion and MVS .	203
12.4.6. Comparison and selection guide	203
12.5. Point clouds and 3D representations	204
12.5.1. Point-cloud data structures and file formats	204
12.5.2. Filtering, downsampling and normal estimation . . .	204
12.5.3. Meshing and surface reconstruction	204
12.5.4. Voxelisation and occupancy representations	204
12.6. Point-cloud registration and alignment	205
12.6.1. Iterative closest point (ICP) and variants	205
12.6.2. Feature-based 3D registration	205
12.6.3. Global alignment strategies	205
12.7. 3D measurement in practice	205
12.7.1. Sensor acceptance testing with calibrated artefacts .	205
12.7.2. Surface interaction: specular and translucent materials	206
12.7.3. Warm-up drift and environmental control	206
13. Motion, tracking and event-based vision	207
13.1. Motion representation	207
13.1.1. The brightness-constancy assumption	207
13.1.2. Sparse vs dense motion fields	207
13.2. Optical flow	208
13.2.1. The Lucas–Kanade method	208
13.2.2. The Horn–Schunck method	208
13.2.3. Dense flow: Farnebäck and pyramidal schemes . . .	208
13.2.4. Learned optical flow (outline)	208
13.3. Background modelling and change detection	209
13.3.1. Frame differencing	209
13.3.2. Gaussian mixture models	209
13.3.3. Robust background subtraction	209
13.4. Object tracking	209
13.4.1. Template and feature tracking	209

13.4.2.	Mean-shift and correlation-filter trackers	210
13.4.3.	Kalman filters and particle filters	210
13.4.4.	Multi-object tracking and data association	210
13.5.	Event-based vision	210
13.5.1.	The event-camera imaging principle	210
13.5.2.	Representations: event frames, time surfaces, voxel grids	211
13.5.3.	Algorithms on event streams	211
13.5.4.	Industrial fit and integration considerations	211

V. Deep Learning 212

14. Machine learning foundations for vision 214

14.1.	The learning paradigm	214
14.1.1.	Supervised, unsupervised and self-supervised learning	214
14.1.2.	Training, validation and test sets	214
14.1.3.	Bias, variance and generalisation	215
14.1.4.	Cross-validation and model selection	215
14.2.	Classical classifiers	215
14.2.1.	Linear and logistic regression	215
14.2.2.	K-nearest neighbours	215
14.2.3.	Support vector machines and kernels	216
14.2.4.	Decision trees, random forests and boosting	216
14.2.5.	Bayesian classifiers	216
14.3.	Feature engineering and dimensionality reduction	216
14.3.1.	Hand-crafted image features	216
14.3.2.	Principal component analysis	217
14.3.3.	Linear discriminant analysis	217
14.3.4.	Manifold learning (outline)	217
14.4.	Loss functions and optimisation	217
14.4.1.	Squared, absolute and Huber loss	217
14.4.2.	Cross-entropy and log-loss	218
14.4.3.	Gradient descent and its variants	218

14.4.4. Regularisation: L1, L2, elastic net	218
14.5. Evaluation of classifiers	218
14.5.1. The confusion matrix	218
14.5.2. Accuracy, precision, recall, F-score	219
14.5.3. ROC and precision-recall curves; AUC	219
14.5.4. Class imbalance and industrial reality	219
14.6. Choosing classical versus deep approaches	219
14.6.1. Data availability and labelling cost	219
14.6.2. Interpretability and validation burden	220
14.6.3. When hand-crafted features still win	220
15. Neural networks and deep learning	221
15.1. Feed-forward networks	221
15.1.1. Neurons, layers and activation functions	221
15.1.2. ReLU, sigmoid, tanh, GELU	221
15.1.3. Softmax outputs and cross-entropy	222
15.1.4. Universal approximation and the role of depth	222
15.2. Backpropagation and automatic differentiation	222
15.2.1. The chain rule on the computational graph	222
15.2.2. Forward and reverse-mode differentiation	222
15.2.3. Practical implementation in modern frameworks	223
15.3. Optimisation of deep models	223
15.3.1. Stochastic gradient descent and mini-batching	223
15.3.2. Momentum, Nesterov and Adam	223
15.3.3. Learning-rate schedules and warm-up	223
15.3.4. Weight decay, dropout, batch and layer normalisation	224
15.3.5. Data augmentation for images	224
15.3.6. Early stopping and validation monitoring	224
15.4. Convolutional neural networks	224
15.4.1. The convolutional layer and parameter sharing	224
15.4.2. Pooling, stride and receptive field	225
15.4.3. Padding conventions and output-size arithmetic	225
15.4.4. Feature hierarchies and interpretability	225

15.5. Canonical architectures	225
15.5.1. LeNet-5, AlexNet, VGG	225
15.5.2. GoogLeNet / Inception	226
15.5.3. ResNet and residual connections	226
15.5.4. DenseNet, MobileNet, EfficientNet	226
15.5.5. Vision transformers and hybrid models	226
15.6. Training strategy for industrial data	227
15.6.1. Transfer learning and fine-tuning	227
15.6.2. Small-data regimes and few-shot learning	227
15.6.3. Class-imbalance countermeasures	227
15.6.4. Reproducibility: seeds, versioning, provenance	227
16. Deep learning for classification, detection, segmentation and anomaly	228
16.1. Image classification	228
16.1.1. Whole-image classification pipelines	228
16.1.2. Multi-label classification	228
16.1.3. Fine-grained classification	229
16.1.4. Evaluation with imbalanced industrial datasets	229
16.2. Object detection	229
16.2.1. Two-stage detectors: R-CNN, Fast R-CNN, Faster R-CNN	229
16.2.2. One-stage detectors: YOLO family, SSD, RetinaNet	229
16.2.3. Anchor-free detectors: FCOS, CenterNet	230
16.2.4. Transformer-based detection: DETR	230
16.2.5. Evaluation: IoU, mAP, per-class analysis	230
16.3. Semantic and instance segmentation	230
16.3.1. Fully convolutional networks	230
16.3.2. U-Net and encoder–decoder architectures	231
16.3.3. DeepLab and dilated convolutions	231
16.3.4. Mask R-CNN for instance segmentation	231
16.3.5. Panoptic segmentation	231

16.4. Anomaly and defect detection	232
16.4.1. Reconstruction-based methods: autoencoders and GANs	232
16.4.2. Embedding-based methods: PatchCore, PaDiM, SPADE	232
16.4.3. One-class classifiers and Deep SVDD	232
16.4.4. Benchmarks: MVTec AD and industrial extensions .	232
16.4.5. Choosing between anomaly detection and supervised approaches	233
16.5. Related tasks in industrial vision	233
16.5.1. Keypoint detection and pose estimation	233
16.5.2. Deep OCR and OCV	233
16.5.3. Depth estimation from a single image	233
16.5.4. Self-supervised representation learning	234

VI. Deployment and Compliance 235

17.From theory to tools: selection, cross-platform mapping and inspection-flow design	237
17.1. The industrial vision software landscape	237
17.1.1. Configurable tool sets versus code libraries	237
17.1.2. Open-source libraries: OpenCV, scikit-image, PyTorch/TF ecosystems	238
17.1.3. Commercial libraries and SDKs: Halcon, VisionPro, Sapera	238
17.1.4. Smart-sensor and no-code environments: Keyence, Zebra Aurora, In-Sight	238
17.1.5. Configuring versus programming: trade-offs and consequences	238
17.2. Cross-platform tool mapping	239
17.2.1. How to read the mapping tables: theory concept to platform tool	239
17.2.2. Acquisition and preprocessing across platforms . . .	239

17.2.3.	Edge, line and circle detection: OpenCV / VisionPro / Halcon / In-Sight / Keyence / Zebra Aurora . . .	239
17.2.4.	Pattern location and alignment across platforms . .	239
17.2.5.	Measurement and gauging tools across platforms . .	240
17.2.6.	Defect and surface-inspection tools across platforms	240
17.2.7.	Identification: barcode, Data Matrix, OCR/OCV across platforms	240
17.2.8.	Deep-learning tools across platforms	240
17.2.9.	Limits of equivalence: where tools do not map cleanly	241
17.3.	Selecting and prioritising tools	241
17.3.1.	From feature and task to candidate tools	241
17.3.2.	Prioritisation: robustness, speed, determinism, maintainability	241
17.3.3.	Rule-based versus learned tools for a given problem	241
17.3.4.	A tool-selection decision guide, cross-referenced to theory chapters	242
17.4.	Designing an inspection as a sequence of steps	242
17.4.1.	Decomposing an objective into ordered steps	242
17.4.2.	The step pattern: objective, tools, region, output . .	242
17.4.3.	Locate-mask-measure-decide as a canonical flow . .	242
17.4.4.	Worked example: scratch inspection decomposed into steps	243
17.5.	Thresholds, checks and pass/fail logic	243
17.5.1.	Per-step checks: presence, contrast, dimension, position limits	243
17.5.2.	Setting limits and thresholds on tool outputs	243
17.5.3.	Building the overall pass/fail decision from step results	243
17.5.4.	Failure attribution: knowing which step failed and why	244
17.6.	Exposing KPIs and data-driven threshold setting	244
17.6.1.	Choosing KPIs and measured outputs per tool and step	244
17.6.2.	Logging tool outputs for offline analysis	244
17.6.3.	Batch statistical analysis to set thresholds and margins	244

17.6.4. Separating good/bad populations and setting operating points	245
17.6.5. Linking threshold setting to the validation of Chapter 18	245
17.7. Software design practice for vision applications	245
17.7.1. Modularity, reuse and configuration management	245
17.7.2. Robustness: handling absent, occluded and out-of-spec parts	245
17.7.3. Defensive checks, error states and safe defaults	246
17.7.4. Version control, documentation and change control of vision jobs	246
17.7.5. Performance, cycle time and resource budgeting	246
18. Systems integration and industrial interfacing	247
18.1. Industrial communication	247
18.1.1. EtherNet/IP and PROFINET	247
18.1.2. Modbus TCP and serial protocols	247
18.1.3. OPC-UA and information modelling	248
18.1.4. MQTT and vision-to-cloud patterns	248
18.2. Integration with PLCs	248
18.2.1. Handshaking and trigger management	248
18.2.2. Result data structures and tags	248
18.2.3. Timing budgets and cycle time	249
18.3. Integration with robots	249
18.3.1. Coordinate-frame transformations and hand-eye calibration	249
18.3.2. Fixed and moving cameras	249
18.3.3. Pick-and-place and vision-guided assembly	249
18.3.4. Bin picking and 3D-guided robotics	250
18.4. Line integration	250
18.4.1. Reject mechanisms and part tracking	250
18.4.2. Statistical process-control feedback	250
18.4.3. Traceability and data-logging architecture	250
18.4.4. Human-machine interfaces for vision stations	251

19. Validation, measurement uncertainty and sampling for inspection	252
19.1. Defining acceptance	252
19.1.1. False accept and false reject rates	252
19.1.2. The confusion matrix in industrial terms	252
19.1.3. Producer and consumer risk	253
19.1.4. Setting acceptance criteria from requirements	253
19.2. Repeatability and reproducibility	253
19.2.1. Sources of variation in a vision measurement	253
19.2.2. Attribute Gauge R&R: method and calculation	253
19.2.3. Variable Gauge R&R by the ANOVA method	254
19.2.4. Kappa statistics and inter-rater/inter-system agree- ment	254
19.2.5. Interpreting %R&R, %tolerance and number of dis- tinct categories	254
19.3. Confidence, tolerance and statistical intervals	254
19.3.1. Confidence intervals for proportions and means	254
19.3.2. Tolerance intervals and coverage	255
19.3.3. Confidence–reliability statements (e.g. 95/95)	255
19.3.4. Propagation of uncertainty through a measurement chain	255
19.4. Sampling plans for inspection validation	255
19.4.1. Attributes versus variables sampling	255
19.4.2. Binomial and hypergeometric bases for attribute plans	256
19.4.3. Operating-characteristic curves, AQL and RQL/LTPD	256
19.4.4. Zero-acceptance ($c=0$) plans and their use	256
19.4.5. Confidence–reliability sample-size formulae	256
19.4.6. ANSI/ASQ Z1.4, Z1.9 and ISO 2859 in practice	257
19.5. Measurement-system analysis for vision	257
19.5.1. Bias, linearity and stability studies	257
19.5.2. Uncertainty budgets and traceability	257
19.5.3. Total measurement uncertainty for vision gauges	257
19.6. Verification, troubleshooting and re-validation	258
19.6.1. A systematic troubleshooting method	258
19.6.2. Isolating optical, lighting, sensor and software failures	258

19.6.3. Recording and root-cause analysis	258
19.6.4. Change control and re-validation triggers	258

20. Deployment, monitoring and human-in-the-loop for learning-based vision 259

20.1. From trained model to production system	259
20.1.1. Model export and interchange (ONNX)	259
20.1.2. Quantisation, pruning and operator fusion	259
20.1.3. Runtime targets: CPU, GPU, edge accelerators	260
20.1.4. Revalidation of the deployed artefact	260
20.2. Dataset governance	260
20.2.1. Collection planning against the operating envelope	260
20.2.2. Labelling protocols and inter-annotator agreement	260
20.2.3. Curation, exclusion records and provenance	261
20.2.4. Versioning and model-to-data traceability	261
20.3. Post-deployment monitoring	261
20.3.1. Domain shift and concept drift	261
20.3.2. Statistical drift detectors	261
20.3.3. Retraining loops and change control	262
20.4. Human-in-the-loop systems	262
20.4.1. Why humans remain in critical and deep-learning loops	262
20.4.2. Review triggers: low confidence, drift, novelty, borderline cases	262
20.4.3. Interface and workflow design for human review	262
20.4.4. Human-in-the-loop as a labelling and improvement engine	263
20.4.5. Levels of autonomy and human oversight	263
20.4.6. Automation bias and the limits of human checking	263
20.5. Explainability and interpretability	263
20.5.1. Saliency and Grad-CAM	263
20.5.2. LIME and local surrogates	264
20.5.3. SHAP and Shapley attributions	264
20.5.4. Limits of post-hoc explanation	264

20.5.5. Explanation as debugging versus explanation for audit	264
20.6. Bias and fairness in industrial vision	265
20.6.1. Subgroups in industrial data: cavities, suppliers, shifts	265
20.6.2. Subgroup analysis and reporting	265
20.6.3. Mitigations: data, loss, imaging	265
20.6.4. Fairness when the system observes people	265
21. Process validation and regulation of vision inspections in regulated industries	266
21.1. Process validation of inspection systems	266
21.1.1. IQ, OQ and PQ for vision systems	266
21.1.2. Validating an inspection against its acceptance criteria	266
21.1.3. Linking validation to the sampling and R&R methods of Chapter 18	267
21.1.4. Software validation and computer-system validation (CSV/CSA)	267
21.1.5. Validating learning-based inspections: special considerations	267
21.2. The EU Artificial Intelligence Act	267
21.2.1. Scope, definitions and territorial reach	267
21.2.2. Prohibited practices	268
21.2.3. The high-risk category and Annex III	268
21.2.4. Obligations for providers and deployers of high-risk AI	268
21.2.5. Limited- and minimal-risk tiers	268
21.2.6. Timelines, enforcement and penalties	269
21.3. US regulation for AI and inspection	269
21.3.1. The FDA framework for AI/ML-enabled devices	269
21.3.2. Predetermined change-control plans and the total-product-lifecycle approach	269
21.3.3. 21 CFR Part 11: electronic records and signatures	269
21.3.4. NIST AI Risk Management Framework	270
21.3.5. Comparison of EU and US approaches	270

21.4. Medical-device and pharmaceutical inspection	270
21.4.1. MDR and IVDR requirements relevant to vision inspection	270
21.4.2. FDA 21 CFR 820 / QSR and process validation expectations	270
21.4.3. GMP and pharma visual-inspection requirements . .	271
21.4.4. Validating automated visual inspection against manual inspection	271
21.5. Other high-risk and safety-critical sectors	271
21.5.1. Aerospace inspection and qualification expectations	271
21.5.2. Automotive quality and functional-safety context . .	271
21.5.3. Common threads across regulated industries	272
Bibliography	273

1. Machine Vision Engineering Handbook

Welcome

The Machine Vision Engineering Handbook is the principal technical knowledge resource of the Machine Vision Academy.

Warning

*In order to make content available as soon as it is developed and reviewed, this handbook has been published with structural placeholders. Individual chapters will be released and marked complete as notated in the **revision history table below**. Chapters with content that has not been marked as released in the Revision History table should not be assumed to be complete nor reviewed.*

Who is this handbook for?

The handbook is intended for:

- engineers designing or specifying industrial vision systems;
- automation and controls engineers integrating vision equipment;
- computer-vision and machine-learning practitioners working in industry;
- students and lecturers studying machine vision;
- quality, validation and regulatory professionals;
- technical reviewers evaluating machine-vision projects.

The material is organised so that readers can use it either as a sequential engineering reference or as a source for individual design, implementation and validation topics.

Revision history

This table records formal content releases of individual sections. A section remains a placeholder until substantive content has been technically reviewed and approved for publication.

Revision	Date	Scope	Status	Notes
Develop- ment skeleton	2026-08-04	Handbook structure	Under development	Initial release of handbook structure

Acknowledgements

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

Notation



Content status: Placeholder.

This subsection has not yet been drafted or technically reviewed.

Part I.

Foundations

i **Part status:** Structural placeholder.
This part introduction will later summarise its scope, prerequisites and learning objectives.

2. Vision engineering: scope, definitions and history

2.1. Machine vision, computer vision and image processing

The terms machine vision, computer vision and image processing overlap, and their boundaries change with professional context. This handbook therefore does not attempt to impose a universal taxonomy. Instead, it establishes a working vocabulary for industrial automation, while recognising the broader usage found in computer science, signal processing and academic research.

2.1.1. Working definitions and industrial success criteria

For this handbook, machine vision is defined as follows:

! Important

“Machine vision is the engineering discipline concerned with specifying, designing, integrating and validating an image-based function that observes a physical process and produces information used for inspection, measurement, identification, guidance or control.”

Its scope can therefore include the definition of requirements and reference truth; part or scene presentation; illumination; optics; sensing and acquisition; image processing and computer-vision algorithms; communications and control interfaces; the resulting process action; and evidence that the complete system performs adequately in production.

This systems view is consistent with industrial treatments of machine vision. The European Machine Vision Association (EMVA) presents a chain extending from handling, lighting and optical image formation through acquisition and processing to classification and process feedback [1]. The VDI/VDE/VDMA 2632 series places requirements specification and acceptance testing explicitly within the engineering lifecycle [2].

 Caution

“A production vision system does not begin when an image reaches memory, and it does not end when an algorithm returns a score.”

Requirements deserve particular attention because a vision system cannot reliably automate a criterion that has never been made explicit. Where an existing human inspection provides the starting point, agreement studies can help establish whether the judgement is sufficiently repeatable to become an automated requirement. The detailed methods for requirements engineering, measurement-system analysis and validation are developed later in this handbook.

Industrial success is therefore judged at system level rather than by an algorithm metric alone. A strong offline classification result cannot compensate for unstable image formation, inadequate cycle time, poor reject timing, unrepresented production variation or an ambiguous acceptance criterion. In practice, performance combines the quality of the decision or measurement with robustness, timing, integration and maintainability.

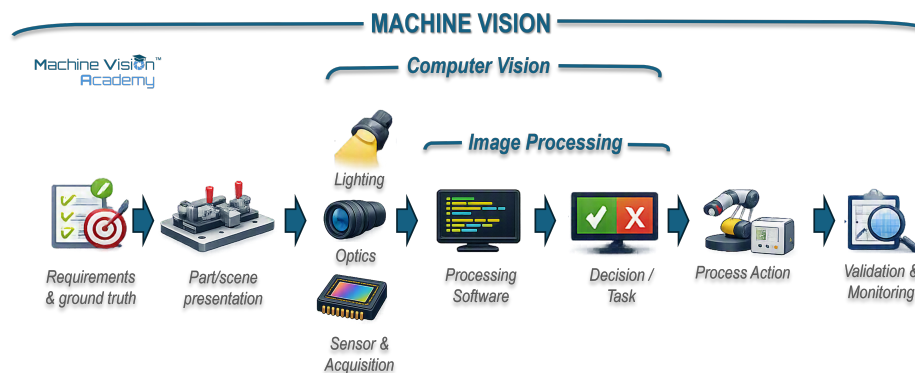


Figure 2.1.: Machine vision as an end-to-end engineering lifecycle

2.1.2. The four fields: image processing, computer vision, machine vision and AI

The following definitions are deliberately operational. They identify the centre of gravity of each field rather than drawing impermeable boundaries. A single industrial application can occupy all four fields at once.

Table 2.1.: Working meanings and typical emphasis of image processing, computer vision, machine vision and artificial intelligence.

Field	Working meaning in this handbook	Typical emphasis
Image processing	Mathematical or computational operations on image data, including correction, enhancement, filtering, transformation, segmentation and representation.	The image and its transformation.
Computer vision	Computational methods that infer information about objects, scenes, geometry, events or states from visual data.	Perception and inference from images.
Machine vision	The engineering of an image-based system that must perform a useful function within a physical or automated process.	The complete requirements-to-action system.

Field	Working meaning in this handbook	Typical emphasis
Artificial intelligence	A broad field concerned with systems that perform capabilities associated with intelligence. In later vision chapters, “AI-based vision” will normally denote data-trained models, particularly machine-learning and deep-learning approaches.	Learned inference and decision methods, while recognising that formal AI is broader than deep learning.

These meanings are not universal. Academic computer vision may include acquisition and image processing within its own scope; image processing is a substantial discipline in signal processing; and machine vision has historically also been used for practical artificial vision more generally. Davies discusses the historical convergence of computer vision and machine vision [3]. The British Machine Vision Association (BMVA) provides a visible example of the overlap: its remit explicitly includes machine vision, image processing and pattern recognition [4].

The same caution applies to AI. ISO/IEC terminology treats artificial intelligence as a broad field and does not equate it with neural networks or deep learning [5]. For clarity in the engineering chapters, the more useful distinction will usually be between classical or explicitly engineered vision methods and learned vision methods. Deep neural networks are a major family of learned methods, but they are not the formal boundary of AI.

i Note

“The labels are useful only when they clarify engineering responsibility. They become misleading when a field name is treated as evidence of a capability that has not actually been demonstrated.”

2.1.3. Relations, overlaps and typical confusions

Terminological overlap matters most when expertise is assessed. A computer-vision specialist may have considerable depth in segmentation, detection, pose estimation, deep learning or open-source algorithm development while having limited experience of illumination, lenses, triggering, industrial communications or reject mechanisms. An experienced machine-vision integrator may show the reverse profile: strong acquisition and production-system engineering, but less depth in advanced computer-vision research. Neither profile is inherently more complete; the useful question is which part of the problem has actually been engineered.

💡 Tip

“When assessing machine-vision expertise, ask which parts of the requirements-to-action chain the person has designed, commissioned and validated—not only which algorithms or products they have used.”

A second confusion arises from the word *train*. Classical pattern-matching systems can require a template image and may expose a command called *Train*, but this does not by itself make them machine-learning systems. Cognex PatMax, for example, represents a trained pattern using geometric features and their spatial relationships [6], [7]. HALCON shape-based matching similarly searches for instances of a shape model that has been

created previously [8]. These are classical model- or template-based techniques rather than neural models fitted to a labelled dataset.

By contrast, learned vision methods estimate model parameters from training data; deep learning is a prominent example. The distinction matters because AI is now also a commercial descriptor. The UK Advertising Standards Authority has documented the widespread use of the term in advertising and the associated regulatory implications [9]. For this handbook, an AI claim is therefore more informative when it states what is learned from data, what model is trained and how performance on unseen data is demonstrated.

The practical relationship is simple enough to carry forward: image processing and computer vision provide computational techniques; machine learning adds families of data-trained inference techniques; and machine vision integrates whichever techniques are appropriate into a physical system that must satisfy an industrial requirement. Later chapters develop acquisition engineering, algorithms, learned vision, integration and validation in detail.

2.2. Historical development: several timelines, one discipline

Machine vision does not have a single birthday. One historical clock started when photographs became arrays of numbers; another when researchers began asking computers to recover shape and meaning from those numbers. A third started on the factory floor, where the decisive question was not whether a machine could understand an arbitrary scene, but whether it could inspect, measure or guide reliably at production speed. Consumer products then made many of the same technologies familiar, while standards emerged to make an increasingly diverse industrial ecosystem interoperable.

Those clocks do not tick at the same rate. A technique can exist in a research paper for years before sensors, processors, software tools and interfaces make it practical in production. Equally, a supposedly new consumer feature may have a surprisingly old industrial ancestor. The history is therefore presented here as a set of parallel timelines rather than a single sequence of inventions. The purpose is orientation: later chapters return to the individual technologies, architectures and validation methods in engineering detail.

2.2.1. The innovation timeline: when images became computable

A convenient starting point is 1957. At the US National Bureau of Standards, now NIST, Russell Kirsch and colleagues used a rotating-drum scanner with the SEAC computer to digitise a photograph of Kirsch's infant son, Walden. The image was only 176 pixels on each side, but it established the essential abstraction behind modern vision engineering: a picture could be represented as numbers and therefore manipulated by a computer [10].



Figure 2.2.: Image of Walden Kirsch, first digital picture 1957

The next step was to do useful work to those numbers. During the Ranger and Surveyor lunar programmes, engineers at NASA's Jet Propulsion Laboratory digitised and processed spacecraft imagery to reduce noise, compensate for distortions and improve visibility. Ranger 7 alone returned 4,308 images in the final minutes before lunar impact on 31 July 1964 [11],

[12]. The lineage from spacecraft image correction to industrial flat-field correction, geometric calibration and computational photography is not direct in every implementation, but the engineering idea is familiar: image formation is imperfect, and computation can make the measurement more useful. Later in this handbook we, cover techniques to enhance images for industrial applications.

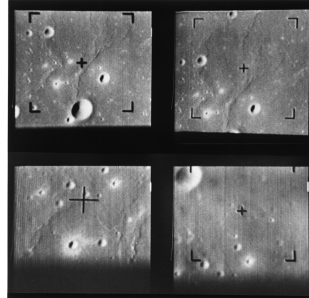


Figure 2.3.: Enhanced images of the moon taken by Ranger 7 - 1964

At roughly the same time, computer vision was becoming a research problem in its own right. Lawrence Roberts' 1963 MIT thesis analysed simple three-dimensional solids from two-dimensional images [13]. In 1966 Seymour Papert wrote the MIT Artificial Intelligence Memo commonly known as the Summer Vision Project, proposing that summer students should connect a camera to a computer and tackle a sequence of visual tasks [14]. The story is often retold as a single optimistic instruction from Marvin Minsky; the surviving proposal is Papert's. Either version captures the mood: researchers had not yet discovered how much of intelligence is hidden inside apparently simple acts of seeing.

The sensing hardware was changing just as quickly. Willard Boyle and George Smith described the charge-coupled device (CCD) in 1970 after its invention at Bell Laboratories in 1969 [15]. Five years later Steven Sasson at Kodak assembled a self-contained solid-state digital still camera using a CCD. The prototype recorded roughly 0.01 megapixels to a cassette and

took about 23 seconds to store an image [16]. It was not a product, but it made a future without photographic film physically demonstrable.



Figure 2.4.: First digital camera prototype developed by Steve Sasson at Kodak (Credit: Getty Images) - 1975

i A 0.01-megapixel glimpse of the future

Sasson's 1975 camera is useful precisely because it looks so unlike a modern camera: about the size of a toaster, with a cassette recorder and a separate playback arrangement. The important innovation was not convenience; it was that the complete image chain - sensor, digitisation, storage and display - had become electronic. An engineering prototype can be historically important long before it is commercially sensible; this one looked like a kitchen appliance and stored its pictures on the same cassette tapes people used for mix-tapes [16]. The prototype was rejected by Kodak in an attempt to protect their dominance in the analog photography business.

Digitisation was only the beginning. From the 1960s to the 1980s, a classical toolbox accumulated that is still recognisable in industrial vision software. Hu’s 1962 moment invariants provided compact shape descriptors designed to remain useful under changes of position, scale and orientation [17]. Rosenfeld and Pfaltz’s 1966 sequential picture operations included connected-component labelling - the basis of what machine-vision engineers still call blob analysis [18]. Otsu’s 1979 threshold-selection method gave a principled way to separate foreground from background [19], while Ballard’s 1981 generalised Hough transform extended voting-based detection from simple analytic curves to arbitrary shapes [20]. Fischler and Bolles’ RANSAC, also published in 1981, made geometric model fitting robust to outliers [21]. Canny’s 1986 edge detector and Harris and Stephens’ 1988 corner detector then supplied local features that became enduring building blocks [22], [23]. The terminology sounds historical; the underlying operations - segment, label, describe, fit, locate and measure - remain everyday machine-vision tasks.

Some of the mathematics behind recognition was older than digital imaging itself. Pearson’s 1901 closest-fit formulation is an origin point for principal-component analysis (PCA) [24]. Much later, Turk and Pentland’s 1991 “eigenfaces” showed how face images could be represented in a much lower-dimensional appearance space [25]. The method is not a direct ancestor of every modern learned embedding, but the engineering instinct is familiar: retain the variation useful to a decision while discarding dimensions that contribute little.

Alongside this practical toolbox, the field was acquiring a language for thinking about vision. David Marr’s posthumous 1982 book framed visual processing at computational, algorithmic/representational and implementation levels [26]. In 1986 Rumelhart, Hinton and Williams published the influential back-propagation paper showing how multilayer networks could learn internal representations by adjusting connection weights [27]. Neither event immediately transformed factory inspection, but both helped define computational and learning frameworks on which later work would build.

A parallel strand asked not only what a machine could see, but what it could do with the result. SRI’s Shakey project (1966-1972) combined a television camera with perception, planning and mobile action [28]; in 1979 the Stanford Cart used stereoscopic vision to cross a chair-strewn room autonomously, although its 20-metre journey took about five hours [29]. In industry, Westinghouse’s 1983 APAS research used machine vision for positioning, orientation and inspection in flexible assembly [30]. Tsai and Lenz’s 1989 hand-eye calibration method provided a practical mathematical bridge between camera and robot coordinate frames [31]; Besl and McKay’s 1992 ICP method became foundational for aligning measured 3D shapes with models [32]; and by 1996 visual-servo research had formalised how image measurements could close the motion-control loop [33]. The recurring systems problem was already clear: a visual result becomes useful to a robot only when it can be converted into geometry and feedback the machine can act upon.

i Twenty metres in five hours

The Stanford Cart is a useful reminder of how quickly expectations move. Its 1979 autonomous run through a room was a landmark in vision-guided navigation, yet the cart stopped roughly every metre while its computers spent ten to fifteen minutes interpreting stereo images and choosing the next move [29]. What reads today like an impractically slow robot was, at the time, evidence that a machine could use vision to plan its own path. It advanced at roughly the pace of a contemplative tortoise; by the standards of 1979, that was astonishing.

Sensor integration then changed the economics. Eric Fossum’s work on CMOS active-pixel sensors at JPL in the early 1990s led to the “camera-on-a-chip” architecture reviewed in his 1997 IEEE paper [34]. Meanwhile, convolutional neural networks were becoming practical for constrained recognition tasks: LeCun and colleagues reported gradient-based document-recognition systems in 1998 [35]. These systems still addressed tightly

framed problems, but they demonstrated that useful visual features could increasingly be learned rather than entirely hand-designed.

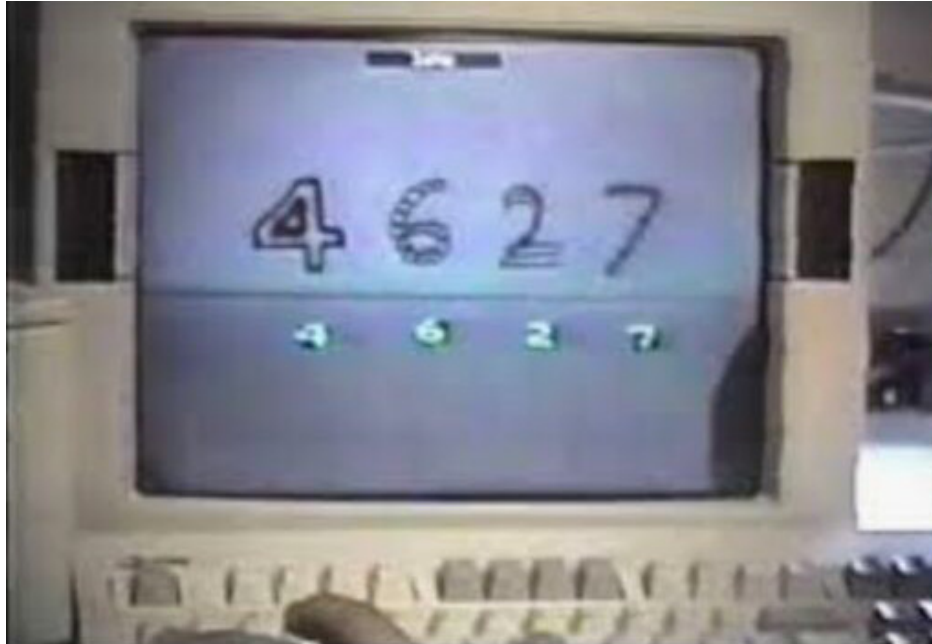


Figure 2.5.: LeCun’s demonstration at Bell Laboratories of a character-recognition system with LeNet - 1998

Hand-engineered recognition was also maturing. Viola and Jones’ 2001 boosted cascade brought fast face detection into real-time products [36], while Lowe’s SIFT formulation in 2004 made local image features much more robust to changes in scale, orientation and viewpoint [37]. This was the mature feature-engineering era: carefully designed representations, geometric consistency and robust matching rather than end-to-end learned features.

The next discontinuity came from scale rather than a single new mathematical idea. ImageNet, introduced in 2009, supplied millions of labelled images

organised around a large semantic hierarchy [38]. In 2012 Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton used a deep convolutional network and GPUs to win the ImageNet challenge by a striking margin [39]. The result accelerated a shift already under way: instead of engineers specifying most visual features by hand, increasingly large parts of the representation could be learned from data. Industrial products would absorb that change more cautiously, because a production line values repeatability, latency, traceability and failure handling as much as benchmark accuracy.

AlexNet was a turning point, not an endpoint. R-CNN showed in 2014 how deep features could drive object detection [40]; U-Net in 2015 established an influential encoder-decoder architecture for dense segmentation [41]; ResNet made substantially deeper models trainable through residual connections [42]; and YOLO reframed object detection as a single real-time network [43]. By 2020-2021, Vision Transformers showed that self-attention could compete with convolution as a core visual representation [44]. Later chapters return to these architectures in detail; here they simply mark how quickly learned vision expanded from classification into localisation, detection, segmentation and more general visual representations.

A parallel software timeline made those methods much easier to reproduce and reuse. Intel’s OpenCV project reached its first public release in June 2000, putting a broad collection of computer-vision routines into an open-source library that could run on ordinary computers [45]. Google open-sourced TensorFlow on 9 November 2015, turning infrastructure developed for large-scale machine learning into a generally available framework [46], [47]. PyTorch development began at Facebook/Meta in 2016 and the framework was released publicly in January 2017 [48], [49]. These libraries did not originate as industrial machine-vision products, but they greatly lowered the barrier between research code and deployable vision software. OpenCV in particular became a common foundation for bespoke computer-vision applications, while TensorFlow and PyTorch accelerated the spread of trained neural models into both research and commercial vision.

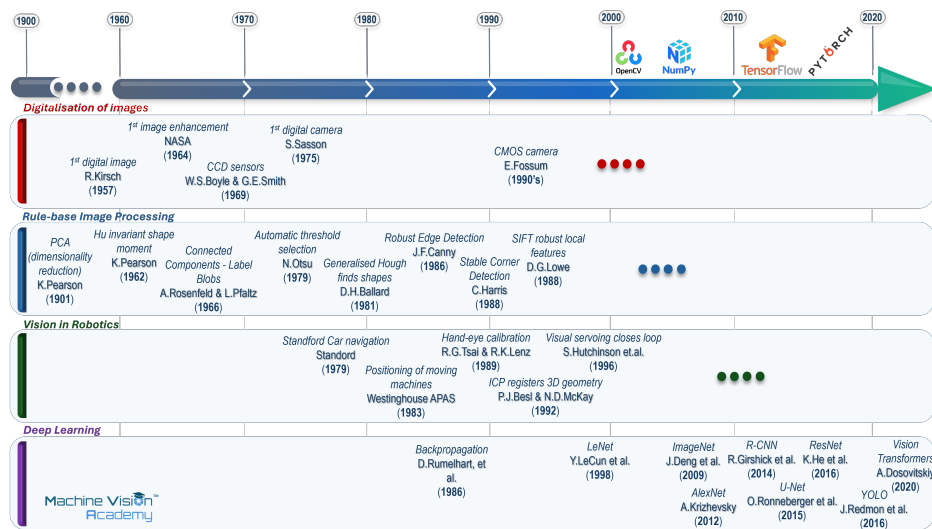


Figure 2.6.: Four parallel tracks describing the innovation timeline: Digit-
alisation, Rule base, Robotics & Deep Learning

2.2.2. The industrial timeline: from bespoke lab prototypes to deployable vision

Industrial machine vision followed a different rhythm from academic computer vision. Early systems of the 1970s and 1980s were often sizeable, expensive and specialised. A3’s historical accounts describe an era in which VME backplanes and dedicated image-processing hardware were commonplace, while suitable cameras were scarce and costly [50]. The important industrial breakthrough was therefore not one algorithm but the gradual availability of an entire stack: reliable solid-state cameras, faster processors, frame grabbers, usable software, industrial I/O and engineers who knew how to make the image itself repeatable.

The creation of the Automated Imaging Association (AIA) in 1984 is a useful marker of that transition from experimental technology to recognisable industry [51]. By the late 1980s and early 1990s, camera suppliers were designing products specifically for machine-vision requirements, personal computers were displacing proprietary processing architectures, and systems were becoming sufficiently reliable for broader deployment. This is the period often remembered within the industry as machine vision’s commercial coming of age.

The 1990s also changed where the computation lived. PC-based systems benefited from commodity processors and flexible expansion hardware; smart cameras moved processing closer to the point of acquisition. A3 notes that the first smart cameras appeared in the 1990s, although the boundary between a “vision sensor” and a “smart camera” has never been perfectly sharp [50]. This architectural choice - central computer, embedded controller or intelligence inside the camera - remains active today, only with much more computing power available in every category.

Standardisation was another quiet revolution. Camera Link, developed in 2000, standardised a high-performance digital connection between cameras and frame grabbers that could be purchased and install in an engineering PC. GigE Vision followed in 2006 and USB3 Vision in 2013, while Camera

Link HS and CoaXPress addressed higher-bandwidth applications [52]. The first module of the EMVA 1288 sensor-characterisation standard was released in 2005 [53], while GenICam, whose working group began in 2003 and issued its first release in 2006, supplied a common software model for camera configuration across different transport technologies [54]. These standards mattered because they reduced ambiguity and the cost of changing components: machine builders could increasingly compare sensors more consistently and select cameras, interfaces and software from different suppliers without redesigning the whole acquisition architecture.

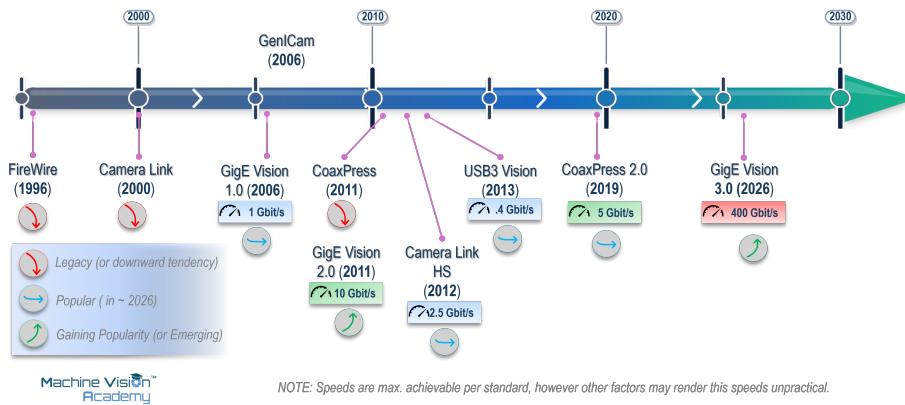


Figure 2.7.: Release timeline of some relevant communication standards for industrial cameras

Three-dimensional sensing then moved from specialist metrology towards mainstream in-line inspection. Laser triangulation profilers, time-of-flight cameras, structured-light systems and later compact 3D smart sensors made height and shape available as ordinary production data rather than laboratory measurements. By the 2010s, products such as LMI's all-in-one Gocator 3110 combined 3D point-cloud acquisition, embedded measurement tools and control decisions in a single sensor [55]. The historical direction is consistent: functions that once required a rack of hardware migrate into smaller, more standardised and more autonomous devices.

The same pattern is now visible with learned vision. Deep learning first changed academic benchmarks, then appeared as server- or GPU-based industrial software, and eventually moved into deployable smart-camera products. Cognex, for example, launched the In-Sight D900 in 2020 and described it as the first industrial smart camera with embedded deep-learning software [56]. OMRON introduced AI-based defect-detection functions into its FH platform in the same period [57]. The industrial adoption curve is slower than the research curve, but often more consequential: once a method survives the requirements of cycle time, maintenance and controlled change, it can be replicated across thousands of production assets.

2.2.3. The product timeline: how the industrial toolbox formed

Industrial machine vision is unusually shaped by product companies. Research establishes what is possible, but suppliers decide how that possibility reaches an engineer: as a camera, frame grabber, library, smart sensor, development environment, 3D profiler or integrated inspection platform. Table 2.2 therefore records representative milestones from companies that have had a visible influence on the industrial toolbox. It is not intended as a ranking or an exhaustive corporate history.

The trouble with claiming a first

Claims of “first” require care. Where the source is a company archive or press release, the wording below treats the claim as the supplier’s claim rather than as an uncontested historical fact. This matters because product categories evolve. A 1991 “smart image sensor”, a 1995 “industrial smart camera” and a later fully programmable smart camera can all plausibly be first under different definitions.

Table 2.2.: Representative milestones in the industrial machine-vision supplier ecosystem

Date	Company	Milestone	Why it mattered / source status
1968	Micro-Epsilon	Company founded; later specialised in precision displacement and profile measurement.	Its 1992 history entry records the launch of its first digital laser-triangulation sensors, followed by laser scanners and 3D inspection systems [58].
1974	KEYENCE	Company founded in Japan with a focus on automation and sensing.	Its corporate timeline marks the 2007 LJ Series 2D laser profiler as a high-precision non-contact measurement milestone [59].

Date	Company	Milestone	Why it mattered / source status
1976	National Instruments	Company founded around computer-based instrumentation.	LabVIEW arrived in 1986; NI later extended the same “software-defined instrument” idea into PC-based machine vision [60].
1976	Matrox	Matrox founded; a dedicated imaging lineage developed alongside graphics and video.	Matrox records the 1986 MVP-AT as the world’s first PC-based image-processing accelerator and the 1993 Matrox Imaging Library (MIL) as an early hardware-independent vision library [61].

Date	Company	Milestone	Why it mattered / source status
1980	DALSA / Teledyne DALSA	DALSA established in Waterloo, Canada.	In the late 1980s DALSA says it released the first commercially available high- performance 4K line-scan image-sensor chip, an important step for high-resolution industrial scanning [62].
1981-1982	Cognex	Cognex founded in 1981; DataMan followed in 1982.	Cognex describes DataMan as the world's first industrial OCR system for direct-marked characters on parts - an early example of vision sold as a production tool rather than a research system [63].

Date	Company	Milestone	Why it mattered / source status
c. 1984	Adept Technologies	AdeptVision developed as an integrated commercial robot-vision system, building on earlier Unim- ation/Machine Intelligence Corporation vision work.	AdeptVision is an early example of vision becoming part of a commercial robot platform rather than a completely separate inspection system. A3 retrospective material describes vision as integrated into the Adept robot controller and records deployment in thousands of systems [64]. The exact product boundary and first-release date are treated as approximate.

Date	Company	Milestone	Why it mattered / source status
1988	Basler	Basler founded.	The company later concentrated on digital industrial cameras; in 2016 it introduced its first 3D camera, a time-of-flight model [65].
Late 1990s	National Instruments	IMAQ Vision integrated machine-vision/image-processing functions with LabVIEW.	Vision became another software-defined engineering task inside a graphical automation environment rather than an isolated vision workstation [60], [66].
1991	DVT	SmartImage Sensor introduced.	DVT representatives later described this as the first smart camera. The claim is useful but definition-dependent [67].

Date	Company	Milestone	Why it mattered / source status
1992	Micro-Epsilon	First digital laser-triangulation sensors in the company timeline.	Shows the parallel growth of optical metrology and machine vision rather than a purely 2D-camera history [58].
1993	Matrox Imaging	Matrox Imaging Library (MIL) introduced.	A long-lived example of the software-library model: one API and toolset abstracting changing acquisition hardware [61].
1995	Vision Components	VC11 presented.	Vision Components describes the DSP-based VC11 as the first industrial smart camera suitable for series use [68].

Date	Company	Milestone	Why it mattered / source status
1996	MVTec	MVTec founded as a spin-off of the Technical University of Munich and FORWISS.	HALCON became a major hardware-independent industrial machine-vision software environment, reinforcing software as a portable layer above cameras and frame grabbers [69].

Date	Company	Milestone	Why it mattered / source status
1998	OMRON	F30 compact machine-vision system introduced.	Contemporary trade coverage describes the F30 as integrating camera, lighting and processor in a compact housing for simple inspection. OMRON also marketed the F10 pattern-matching sensor in this period, but the F10 used a sensor head plus amplifier/processing unit; calling both products single-body smart cameras would therefore be misleading [70], [71].

Date	Company	Milestone	Why it mattered / source status
2000	Cognex	In-Sight family introduced; In-Sight 2000 announced in March 2000.	Cognex entered the low-cost factory-floor vision-sensor market with a spreadsheet-configured system that did not require a PC for operation. Cognex annual reports confirm the In-Sight family was introduced in 2000 [72], [73]. This is a major Cognex smart/embedded-vision milestone, but not an uncontested claim to the first smart camera.

Date	Company	Milestone	Why it mattered / source status
2002	Cognex	VisionPro software development environment introduced.	Cognex's 2002 Form 10-K states that VisionPro was introduced in early 2002 as an ActiveX-based environment exposing Cognex core vision tools in a PC/Microsoft environment [74]. The often-repeated 1998 date for a CVL-to-VisionPro transition was not supported by the primary material reviewed, so it is not used here.

Date	Company	Milestone	Why it mattered / source status
c. 2004	KUKA / Cognex	KUKA.Vision for the KRC2 generation provided an integrated route for using Cognex In-Sight smart cameras with KUKA robots.	The exact first-release year has not been independently established in the sources reviewed. Cognex documentation confirms KUKA KR C2 and KR C2 Edition 2005 support and a KUKA-specific In-Sight communication mode [75]. Historically, the integration illustrates a partnership strategy in which a robot manufacturer adopted a specialist vision supplier rather than developing the complete vision stack internally.

Date	Company	Milestone	Why it mattered / source status
2005	Cognex / DVT	Cognex completed its acquisition of DVT Corporation on 9 May 2005.	DVT was a significant supplier of low-cost vision sensors and had an established worldwide distributor network.
2005	Matrox Imaging	Iris P-Series programmable smart cameras expanded.	Contemporary coverage records the Iris P300H in July 2005 and remote-head P-Series models in September. The cameras combined image sensing, embedded PC-class processing, Windows CE .NET and the Matrox Imaging Library API [76], [77].

Date	Company	Milestone	Why it mattered / source status
2006	FANUC Robotics	iRVision introduced as FANUC's built-in robot-vision package for the R-J3iC controller family.	A contemporary FANUC announcement described iRVision as the company's first built-in vision package for the R-J3iC family, requiring only a camera and cable rather than separate processing hardware [78]. It represents the vertically integrated route: robot manufacturer, controller and vision environment supplied as one platform.

Date	Company	Milestone	Why it mattered / source status
2007	KEYENCE	LJ Series 2D laser profiler launched.	Representative of laser profiling moving into accessible, in-line non-contact measurement [59].
2007-2008	KEYENCE	CV-5000 generation of controller-based vision systems entered service.	The CV-5000/CV-3000 families are controller-based vision systems rather than self-contained smart cameras. VS Series will later offer smart-camera formats [79].
2011	Teledyne / DALSA	Teledyne completed its acquisition of DALSA.	A marker of consolidation around portfolios spanning sensors, cameras, acquisition and imaging components [80].

Date	Company	Milestone	Why it mattered / source status
c. 2012-2013	KUKA / Cognex	KUKA.VisionTech brought Cognex VisionPro-based 2D vision functionality into the KUKA robot environment for localisation, inspection and code/OCR tasks.	Cognex identifies KUKA.VisionTech as integrating VisionPro software with KUKA robots [81]. A KUKA VisionTech 2.1 manual is dated October 2013 [82]; the initial release date is therefore shown approximately. It illustrates deep integration of a specialist third-party machine-vision stack with a robot manufacturer's platform.

Date	Company	Milestone	Why it mattered / source status
2013	LMI Technologies	Gocator 3110 launched.	LMI described it as the industry's first all-in-one 3D smart snapshot sensor, combining acquisition, measurement and control decisions [55].
2016	Basler	First Basler 3D camera, using time-of-flight technology.	Shows industrial- camera suppliers expanding beyond conventional 2D acquisition into depth sensing [65].

Date	Company	Milestone	Why it mattered / source status
2017	Cognex / ViDi Systems	Cognex acquired ViDi Systems SA on 4 April 2017.	The acquisition brought deep-learning software developed specifically for industrial machine vision into Cognex's Vision Products business and was an important step in adding learned inspection to its commercial portfolio [83].
2018	OMRON	FHV7 smart camera announced.	OMRON highlighted an integrated multi-colour light and high-resolution sensor as a way to make one camera adapt to high-mix inspection [84].

Date	Company	Milestone	Why it mattered / source status
2020	Cognex	In-Sight D900 deep-learning smart camera launched.	Cognex claimed the first industrial smart camera with deep learning embedded in the device [56].
2020	OMRON	FH Series gained AI defect-detection capability.	OMRON described the release as an industry-first defect-detection AI for its vision platform; the “first” is a vendor claim [57].
2022	Zebra / Matrox Imaging	Zebra completed the acquisition of Matrox Imaging.	The transaction brought Matrox’s frame grabbers, machine-vision software and systems into Zebra’s industrial automation portfolio [85].

🔥 Who invented the smart camera? It depends what counts

DVT literature dates its SmartImage Sensor to 1991 and describes it as the first smart camera [67]. Vision Components dates the VC11 to 1995 and describes it as the first industrial, series-ready smart camera [68]. A3 itself notes that the border between smart cameras and vision sensors has long been grey [50]. The apparent contradiction is historically useful: “first” depends on whether the defining feature is onboard processing, stand-alone operation, programmability, industrial I/O, or readiness for series deployment. In this chapter, contested firsts are therefore attributed rather than presented as universal facts.

Seen together, these product histories show several different routes by which machine vision matured. NI and MVTec helped make vision programmable and reusable through systems, libraries and development environments. KEYENCE and OMRON pushed highly integrated sensors and smart cameras towards users who might not be vision specialists. Cognex and Matrox (now part of Zebra) have pursued both product strategies, PC based libraries and integrated-edge devices, with increased focus on smart cameras in recent years. DALSA and Basler illustrate the camera and sensor lineage, while LMI and Micro-Epsilon show how optical metrology and 3D profiling became part of the same industrial ecosystem. Robot manufacturers also brought vision closer to the automation platform: FANUC introduced its controller-integrated iRVision in 2006 [78], while KUKA’s collaboration with Cognex is visible in KRC2-era In-Sight integration and later in KUKA.VisionTech around 2012-2013, which integrated Cognex ecosystem into the robot environment [75], [81]. These different approaches - proprietary integrated vision, specialist third-party software and increasingly autonomous smart cameras - illustrate how the boundaries between vision, robotics and automation have steadily blurred as suppliers have added software, AI, 3D and embedded processing.

i Robot vision predates these product milestones

Robot-integrated vision predates the commercial products listed here. The Adept, KUKA and FANUC entries should therefore be read as representative approaches to packaging vision within robot automation platforms, not as claims for the first vision-guided industrial robot.

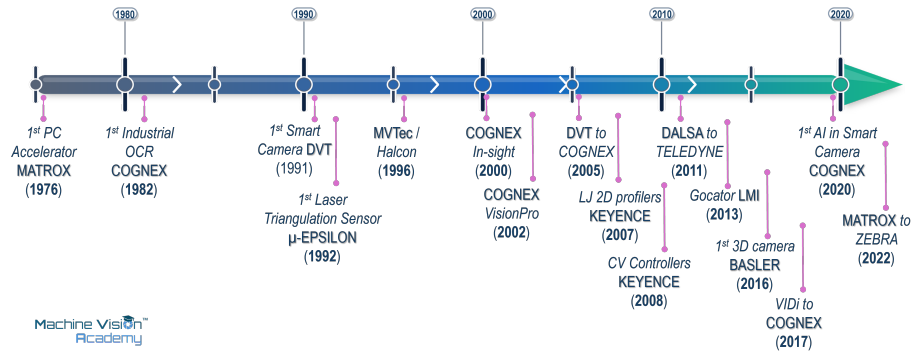


Figure 2.8.: From component to appliance: representative industrial machine-vision product milestones

2.2.4. The everyday consumer timeline: when vision escaped the laboratory and factory

Some of the best historical anchors to get a true sense of the timescale are ordinary everyday consumer devices or services. On 26 June 1974, the first installation of UPC supermarket scanners entered service at a Marsh supermarket in Troy, Ohio. A packet of Wrigley’s chewing gum became the first purchase made with the new scanner system [86]. The system did not “understand” groceries; it read a deliberately designed visual code with exceptional reliability. That is an industrial lesson in miniature: when the environment and target can be engineered, a narrow visual task can

become commercially transformative.

The same inversion appears in the history of the QR Code. DENSO WAVE introduced it in 1994 for manufacturing and logistics, where conventional barcodes were becoming too limited for the amount and speed of information required. Position-detection patterns at three corners made the symbol easy to locate and orient. DENSO WAVE then promoted open use of the standardised code, helping a factory identification technology migrate into tickets, payments and - eventually - restaurant menus [87].

i From a Toyota-group factory problem to the cafe table

The QR Code is a useful reminder that consumer adoption can come long after industrial invention. DENSO WAVE developed the code in 1994 for production and logistics; camera-equipped phones helped it spread in the 2000s, and later mass use made the square pattern one of the most recognisable pieces of machine-readable imagery in everyday life. A code engineered to survive the grime of a car-parts plant turned out to be equally at home surviving a laminated menu [87].

A less consequential but more charming example came from Cambridge. In late 1991, researchers in the University of Cambridge Computer Laboratory pointed a camera at the Trojan Room coffee pot so colleagues elsewhere in the building could avoid a wasted trip when it was empty. The local network application was connected to the World Wide Web in 1993 and became the first webcam .. and the first reality TV shows [88]. The historical significance is disproportionate to the problem: remote visual sensing had become a service rather than a laboratory experiment.

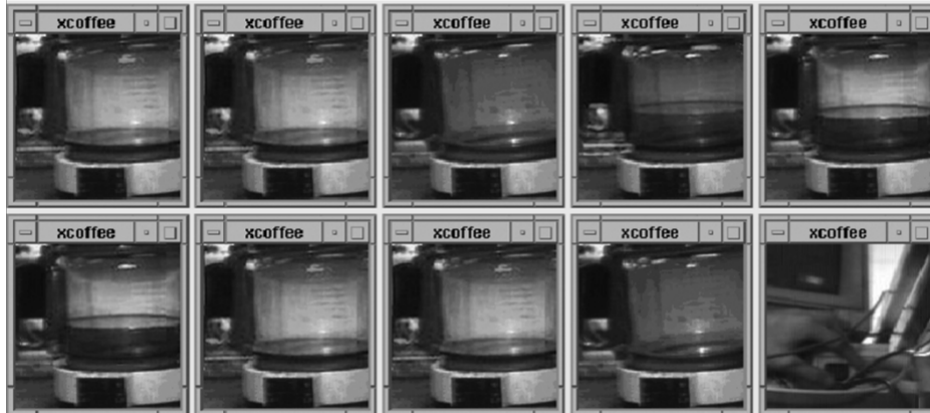


Figure 2.9.: The Trojan Room coffee pot: subject of the first webcam (1991), connected to the World Wide Web in 1993

i The first webcam monitored caffeine, not people

The Trojan Room system is a good antidote to technological grand narratives. The first webcam was not designed for telepresence, social media or surveillance; it solved what may be the most relatable problem in the history of computer vision: sparing researchers the walk across a building only to find the pot empty. It was switched off in August 2001, after a decade of faithful service [88].

By the 2000s, visual computation was increasingly disappearing inside everyday products. Fast face detectors allowed compact cameras to draw focus boxes around faces, while embedded image processing began correcting photographs before the user even saw them.

In 2007 real-time flash-eye detection was introduced in smart phones' firmware by Photonation [89]. For many readers, particularly those born in the new millennium, red-eye may be an unfamiliar photographic artefact precisely because its detection and correction have become so commonplace.

For earlier generations, however, the unintended vampire look was a familiar feature of the family photo album. Developments such as these marked a subtle but important transition: the camera was becoming more than a sensor; it was beginning to interpret and modify its own images.

Kinect brought a different kind of machine vision into the living room. Microsoft launched Kinect for Xbox 360 on 4 November 2010, allowing users to control games through body movement and voice [90]. Guinness World Records records eight million units sold in its first 60 days, making it the fastest-selling gaming peripheral at the time [91]. For millions of people, depth sensing and body tracking arrived not as a robotics demonstration or a metrology system, but as the instruction to stand in front of a television and move.

By the middle of the next decade, semantic vision had become an almost invisible service. Google Photos launched in 2015 with automatic organisation by people, places and things, without requiring users to tag every image manually [92]. Two years later Apple's iPhone X made 3D face recognition an everyday authentication gesture: Face ID combined a dot projector, infrared camera and flood illuminator, projecting more than 30,000 invisible infrared dots and using neural networks to build a mathematical representation of the face [93]. These features are easy to take for granted precisely because the vision system is no longer presented as a separate application; it is simply part of using the product.

In medicine, the same broad progression towards automated visual decision-making reached a much more consequential threshold in 2018, when IDx-DR was authorised in the United States as an autonomous AI-based diagnostic system for diabetic retinopathy. Its pivotal study compared autonomous analysis of retinal photographs with a specialist reference standard [94]. The mechanisms behind a supermarket scanner, automatic red-eye correction, body tracking, face authentication and retinal diagnosis are very different, but the systems pattern is recognisable: capture an image, extract useful information, make a decision and determine what happens next.

2.2.5. What the history reveals

Viewed as parallel timelines, the history is less a sequence of replacements than a sequence of layers. Digital image processing did not disappear when computer vision arrived; classical geometry did not disappear when statistical learning became fashionable; PC-based architectures did not disappear when smart cameras became practical. Industrial systems continue to combine techniques from different decades because the correct choice is governed by the task, not by historical novelty.

A second pattern is migration. The same enabling technology repeatedly crosses domains: space programmes helped advance digital image enhancement and CMOS sensing; factory identification produced the QR Code that later became a consumer interface; research in learned representations eventually became a tool for production defect inspection. The path rarely runs in one direction from university to factory to consumer. Vision engineering is an ecosystem in which ideas, components and constraints circulate.

The third pattern is that industrial progress is often less dramatic than research progress and more dependent on integration. A better detector matters, but so do a stable light, a faster interface, a camera that can be replaced without rewriting software, a 3D sensor that performs its own measurement, and a validation method that explains what “good enough” means. This is why the history of machine vision cannot be told only through algorithms. It is equally the history of sensors, optics, computing architectures, software tools, standards, companies and the production problems that forced them to work together.

! The recurring pattern behind the milestones

Vision technology becomes transformative when three things coincide: a useful visual method, hardware inexpensive and reliable enough to deploy, and an engineering environment that makes the method

repeatable. Research changes what can be imagined; product engineering changes what can be installed; standards and validation change what can be scaled and trusted.

2.3. Industrial application families

The previous chapter established what machine vision is and how the discipline developed. In this chapter we address what machine vision is for. On a factory floor, a technology is valuable because it solves an industrial task reliably, at production speed and, in some sectors, within a regulated quality system. This section is therefore organised around tasks rather than technologies.

A large proportion of industrial vision applications can be grouped into four families: *measurement*, *defect inspection*, *identification* and *guidance*. *Measurement* returns a dimension or other quantitative value against a stated tolerance. *Inspection* determines whether a part is complete, correctly assembled and free of unacceptable defects. *Identification* reads a code or character string and associates the item with a record. *Guidance* locates a feature or object and provides coordinates that an automated system can act upon. Many deployed systems combine two or more of these functions in sequence.

The four application families considered here deliberately reflect the industrial focus of this handbook; they do not represent the full scope of machine and computer vision. Important neighbouring domains include *biometrics*, such as face, fingerprint and iris recognition; *medical imaging and computer-aided diagnosis*; *advanced driver-assistance and autonomous systems*, where visual perception supports functions such as lane keeping, object and pedestrian detection and navigation; and *remote sensing and autonomous robotics*, where visual information is used to interpret and act within less controlled environments. These fields

are outside the principal scope of this handbook, but their development remains relevant to industrial vision. As the historical development of the discipline has already shown, technological progress is rarely linear or confined to one application domain: advances in sensors, learned representations, three-dimensional perception, real-time processing and validation frequently migrate between research, consumer, medical, automotive and industrial applications. Technologies being developed in these neighbouring fields can therefore be expected to influence the future capabilities and architectures of industrial automation.

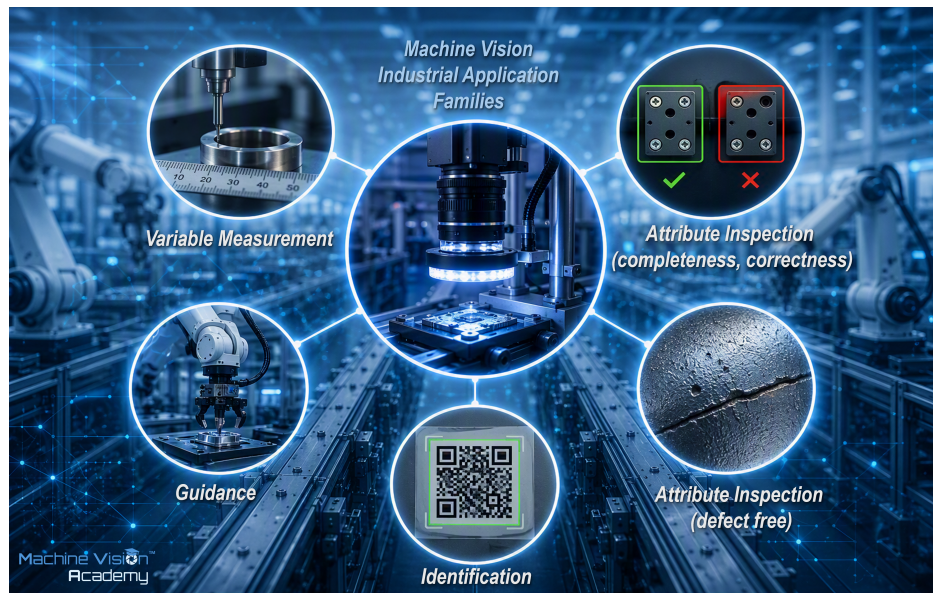


Figure 2.10.: Machine vision industrial application families

The motivation for automating visual inspection in the factory floor is long-standing. Manual visual inspection is repetitive, can be affected by fatigue and variation between inspectors, is difficult to document consistently, and may not keep pace with high-throughput production. Automated visual inspection can apply a defined decision process consistently under

controlled conditions and can retain a record of its measurements and decisions [95].

An alternative but complementary classification, borrowed from quality engineering, is useful because it often points towards the appropriate measurement and decision strategy: **inspection by variables** and **inspection by attributes**. ISO 3951 describes sampling procedures for inspection by variables, whereas ISO 2859 addresses inspection by attributes [96], [97].

A *variable inspection* produces a quantitative result on a scale—for example, a diameter, gap or angle—that can be compared with a specification limit and verified against a calibrated reference. The measurement tasks in Figure 2.10 commonly have this form. They are a natural fit for deterministic, rule-based vision: an edge is located, a distance is calculated, a calibration is applied and the result is compared with a tolerance.

An *attribute inspection* produces a categorical result such as present/absent, conforming/non-conforming or scratched/clean. The underlying decision can still be derived from quantitative measurements; what distinguishes the attribute case is the categorical result used for acceptance. Some attribute checks are readily implemented with hand-crafted features and explicit thresholds. Others involve appearance criteria that are difficult to express analytically and may benefit from labelled examples and learned models. The choice between rule-based and learned methods therefore depends on the nature of the feature, the available data and the required evidence, rather than on the attribute/variable label alone.

i Is it an attribute? Is it a variable? Is it both

Where practical, a variable output is preferable because it preserves information beyond pass or fail. A measured value allows acceptance thresholds to be revised and, more importantly, supports statistical analysis and process monitoring: a population can drift towards a limit long before the reject rate changes.

The difficulty is establishing a meaningful scale. A variable measure-

ment should ideally be calibratable against a recognised reference, such as millimetres or degrees; derived image features such as contrast or texture scores may be much harder to calibrate meaningfully. A useful variable acceptance criterion also requires process characterisation to establish how values on that scale relate to product performance. This is good engineering practice, but it requires experimental evidence and is not always practical.

In reality, the machine-vision engineer often inherits an inspection already defined as an attribute judgement, even when a variable formulation might have been possible.

2.3.1. Measurement and non-contact gauging

Measurement is the family that produces a quantitative value—a length, diameter, angle, gap or height—and a principal advantage of vision-based gauging is that it can be non-contact. A traditional coordinate-measuring machine obtains dimensions by probing a part physically; an imaging system obtains information optically. This can be advantageous for delicate, sterile or soft parts and can permit in-line measurement of every item rather than off-line sampling. The resulting measurements can also feed statistical process control, turning inspection into a continuous source of process information.

Accurate gauging depends strongly on image formation. Telecentric optics can reduce magnification changes caused by object-distance variation; back-lighting can create a high-contrast silhouette; and sub-pixel edge localisation can estimate edge position more finely than the pixel sampling interval. These techniques are among the considerations that distinguish a traceable measurement system from one that merely displays a precise-looking number [98].

 A measured number is not the same as a true one

A vision gauge initially derives a value from image coordinates; calibration against a traceable reference establishes the relationship to physical units, while measurement-uncertainty analysis determines how confidently the result can be used. A non-contact optical measurement and a contact CMM measurement of the same nominal feature can differ because the two systems interact with and sample the part differently. ISO 10360-7 specifies acceptance and reverification tests for Cartesian CMMs equipped with imaging probing systems operating in discrete-point probing mode [99]. Repeatability, reproducibility, bias and uncertainty are therefore as important as the nominal image resolution and would be covered in later chapters.

The measurement toolkit is predominantly deterministic: caliper and edge-pair gauges, line and circle fitting, blob geometry and sub-pixel contour analysis in two dimensions; and laser triangulation, profilometry, structured light and other optical ranging methods for height and shape. Learned methods can assist some localisation or interpretation tasks, but precise dimensional gauging still depends on calibration, traceability and a defined measurement model; industry guidance likewise distinguishes this from appearance-based AI inspection [100]. This is the clearest application of the inspection-by-variables principle [96].

2.3.2. Defect inspection

Defect inspection is the family most commonly associated with machine vision on a factory floor. Typical tasks include *completeness* (e.g. presence of components), *correctness* (e.g. position and orientation, material or variant verification) and *cosmetic defect inspection* (e.g. surface condition and texture, colour, presence of foreign material). What ties these tasks together is a decision on which the process must act within the available

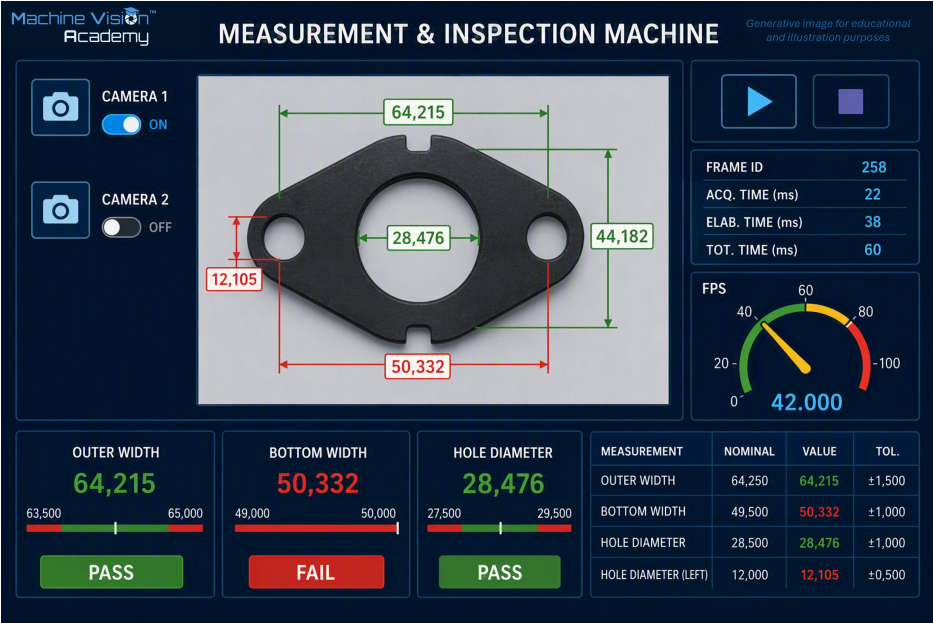


Figure 2.11.: Application example: Non-contact gauging for dimensional inspection

cycle time, for example by rejecting the part or intervening to correct the fault.

Most tasks in this family are attribute checks, although an attribute inspection can contain variable sub-steps. A colour check, for example, may calculate a histogram or colour-distance value on a continuous scale and then convert that value into a categorical decision. The family describes the decision delivered by the inspection, not every mathematical operation used internally.

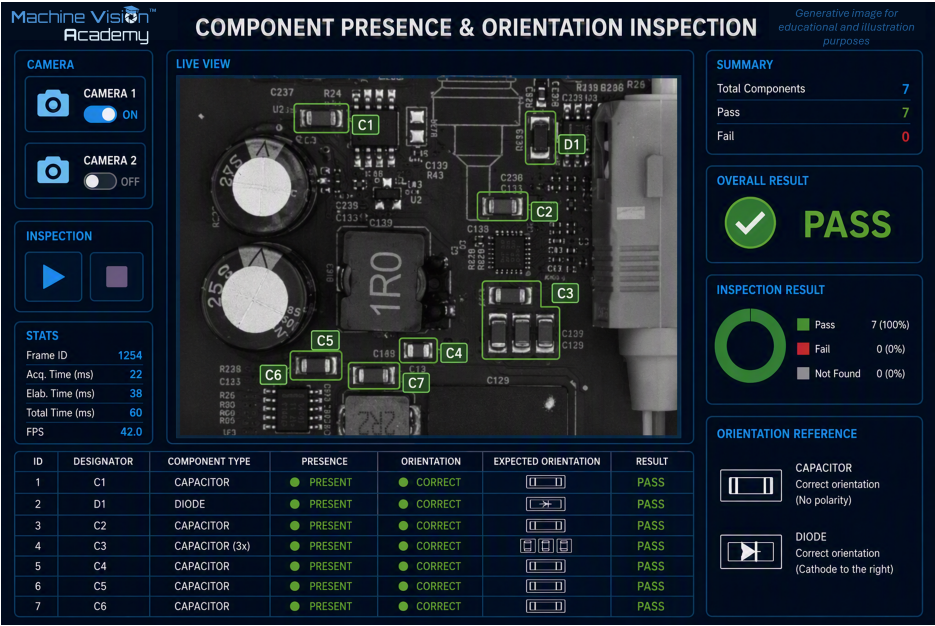


Figure 2.12.: Application example: Component presence and orientation inspection

The traditional approach to implementing these inspections is to translate the acceptance criterion into explicitly engineered image features and decision logic. Typical rule-based tools include thresholding and connected-component analysis, template and geometric pattern matching, edge and

contour analysis, and classical texture or colour measures. These methods are particularly effective when the characteristics that distinguish acceptable from defective parts can be expressed consistently through a manageable set of measurable image features. They are deterministic, generally require relatively few examples during development, and make the relationship between the observed feature and the resulting decision explicit. Their limitations become apparent when acceptable product appearance and defect variation are too complex to describe reliably through hand-crafted features and thresholds.

Deep Learning approaches provide an alternative when that visual variation is difficult to describe explicitly. Depending on the inspection task, these include image classification, object detection, semantic or instance segmentation and anomaly detection. Their flexibility, however, shifts a significant part of the engineering problem from feature design to data: representative examples must be collected, labelled and shown to cover the relevant production variation. This can be particularly difficult in manufacturing because genuine defects are often rare, resulting in small and highly imbalanced datasets. Published work on sterile-barrier seal inspection illustrates this problem and the resulting need to evaluate model behaviour carefully under such conditions [101], [102]. Scarcity of defective samples and class imbalance increase the risk of overfitting and make validation particularly demanding, especially in regulated industries. Rule-based and learned approaches are developed in detail later in the handbook; here they should be regarded as complementary strategies whose suitability depends on the inspection requirement and the available evidence.

2.3.3. Identification of unique parts: codes, characters and traceability

Identification answers the question: *which item is this, and can its history be traced?* It reads machine-readable and human-readable marks—one- and two-dimensional codes, printed text and direct-part marks—and associates

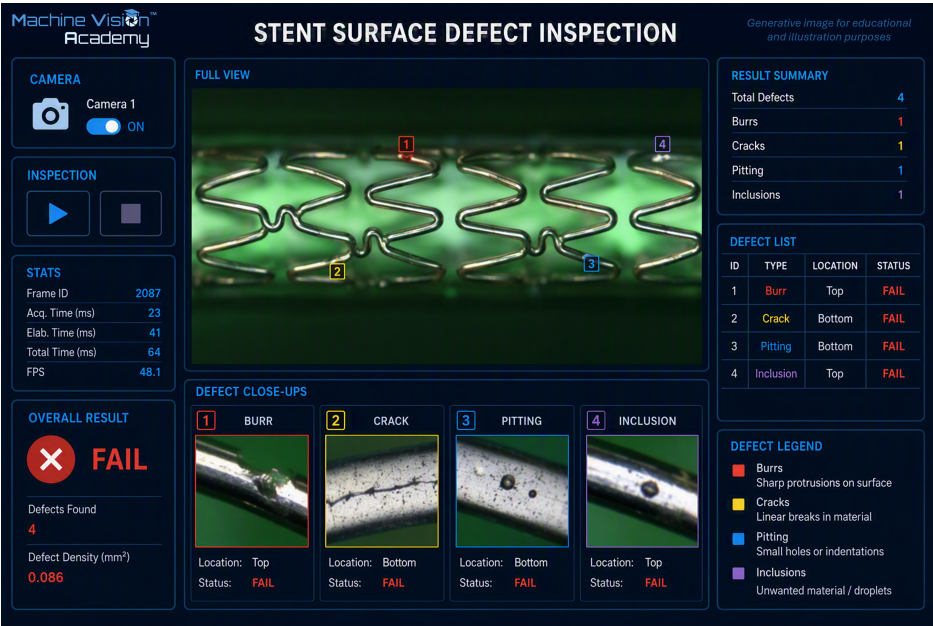


Figure 2.13.: Application example: Surface inspection

each item with a record. In regulated industries this capability supports traceability, authentication and recall processes. Medical-device Unique Device Identification and pharmaceutical serialisation are examples of regulatory contexts in which machine-readable identifiers and verification are important [103], [104]. Industry accounts also emphasise the value of integrating code reading and logging directly into automated production [105].

Data Matrix, standardised by ISO/IEC 16022, is widely used where a compact two-dimensional identifier is required. Direct-part marking by laser, dot-peen, inkjet or other methods can provide a persistent identifier without a separate label. Reading low-contrast marks on curved, reflective, damaged or contaminated surfaces can require full machine-vision techniques rather than a simple idealised barcode-reading scenario [104].

A related pair of tasks is optical character recognition (OCR) and optical character verification (OCV). OCR determines text whose content is not known in advance, whereas OCV checks that an expected string has been produced correctly and legibly [106]. A traceability station can combine these functions: verify printed text, decode a machine-readable identifier, compare the two and reject a mismatch.

The practical difficulty of identification is therefore robustness and speed. Industrial marks may suffer from low contrast, glare, curvature, print variation or direct-marking distortion. Code quality can itself be graded so that the process controls not merely whether a symbol is readable at the current station, but whether its quality is sufficient for downstream use.

2.3.4. Guidance and robot vision

A conventionally taught robot repeats positions defined relative to its coordinate system. If the workpiece is not presented at the expected pose, the operation can fail. Vision guidance reduces that dependence on precise fixturing by locating a feature or object in two or three dimensions and

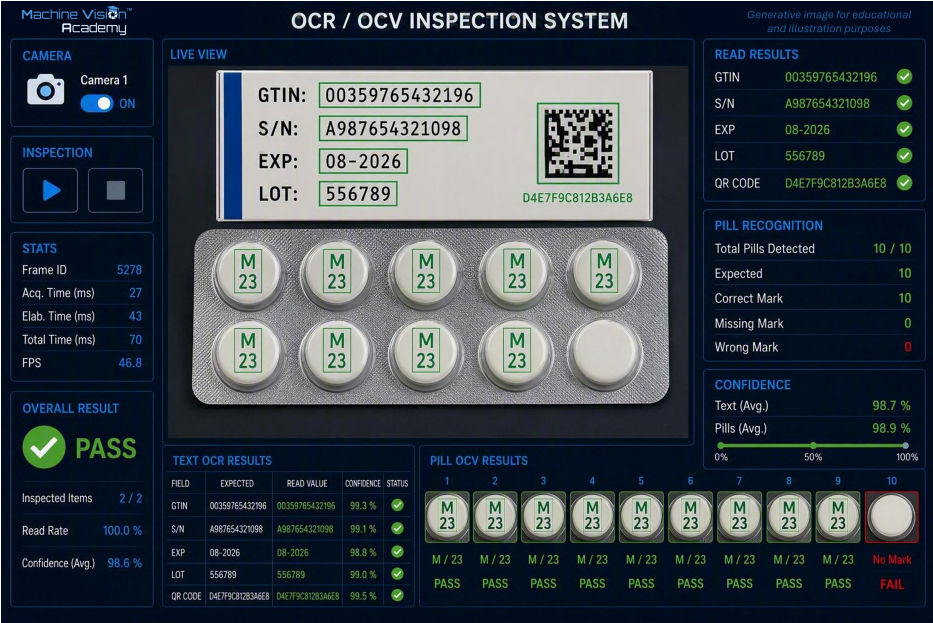


Figure 2.14.: Application example: Product identification and OCR inspection

providing pose information that the robot or motion system can use. The industrial value is flexibility: components can tolerate greater variation in presentation, which can reduce dedicated fixturing and support higher product mix.

Two-dimensional guidance estimates position and in-plane rotation when the relevant geometry can be represented on a plane, for example locating a component on a conveyor or determining an offset for placement. Three-dimensional guidance adds depth and full spatial pose. A representative application is bin picking, in which the system identifies candidate parts in a container, estimates their poses and supports selection of a feasible pick [107].

The difficult engineering problems extend beyond recognition. Reflective, dark or transparent surfaces can challenge depth acquisition; parts may overlap and occlude one another; the complete perception-and-motion sequence must meet cycle time; and the camera coordinate system must be related accurately to the robot coordinate system. Hand-eye calibration is therefore fundamental. Relevant methods include two-dimensional pattern location and pose estimation, three-dimensional acquisition using structured light, time-of-flight or stereo, point-cloud registration, and learned grasp-point selection where object variation makes explicit modelling difficult.

2.4. System architectures

Having looked at what vision systems are asked to do, the next question is where the image-to-decision process should live. This is partly a development question - which tools, skills and computing resources are needed to engineer the solution - but it is primarily a deployment question. Production architecture must satisfy cycle time, physical space, integration, reliability, cybersecurity and long-term maintainability while supporting the availability, performance and quality targets that ultimately contribute to OEE.

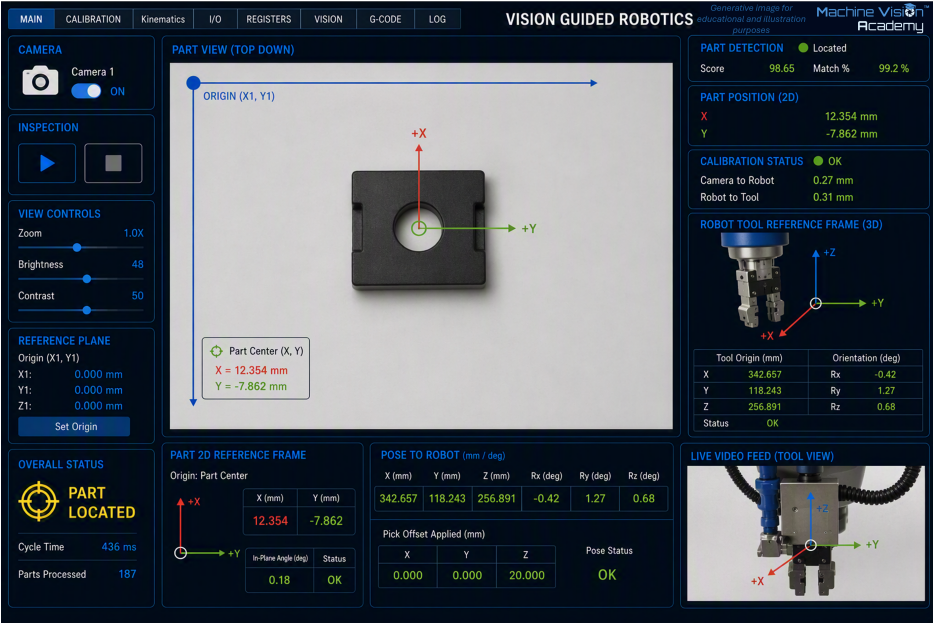


Figure 2.15.: Application example: Robotic guided part picking

Whatever the hardware arrangement, the logical chain remains familiar: **part presentation -> lighting -> optics -> sensor/image acquisition -> image processing -> decision -> process action**, as illustrated in Figure 2.1 and discussed in later chapters. Architectures differ less in what the vision system is trying to achieve than in where processing occurs, how the processing is packaged, what other systems it depends upon and how widely failures can propagate.

A useful way to visualise deployment is as a spectrum extending outwards from the camera. **Device Edge Computing** keeps acquisition and processing inside the imaging device. **Machine Edge Computing** separates the camera from a controller, industrial PC or specialised AI computer located at or beside the machine. **On-Premises Vision Server** centralises processing within the organisation, usually so that compute, storage or software services can be shared. **Cloud Inference** moves the production inference itself to remotely hosted infrastructure. GPUs, NPUs and other accelerators are not separate positions on this spectrum; they are computational choices that can appear at several of these levels.

Hybrid architectures cut across this spectrum in two ways. Production functions can be divided between locations - for example, local inference with central storage and monitoring - and development can be separated from deployment, such as training a model in the cloud and deploying it later to a smart camera or industrial PC. The sections that follow therefore work from the camera outwards, before comparing the principal trade-offs and offering practical selection guidance.

2.4.1. Device Edge Computing

Device Edge Computing places image acquisition, processing and decision-making within the imaging device itself. In machine vision this is represented mainly by the **smart camera, smart vision sensor or smart 3D sensor**: a compact industrial device containing the sensor, processor, memory, vision runtime and communications interfaces, often with digital

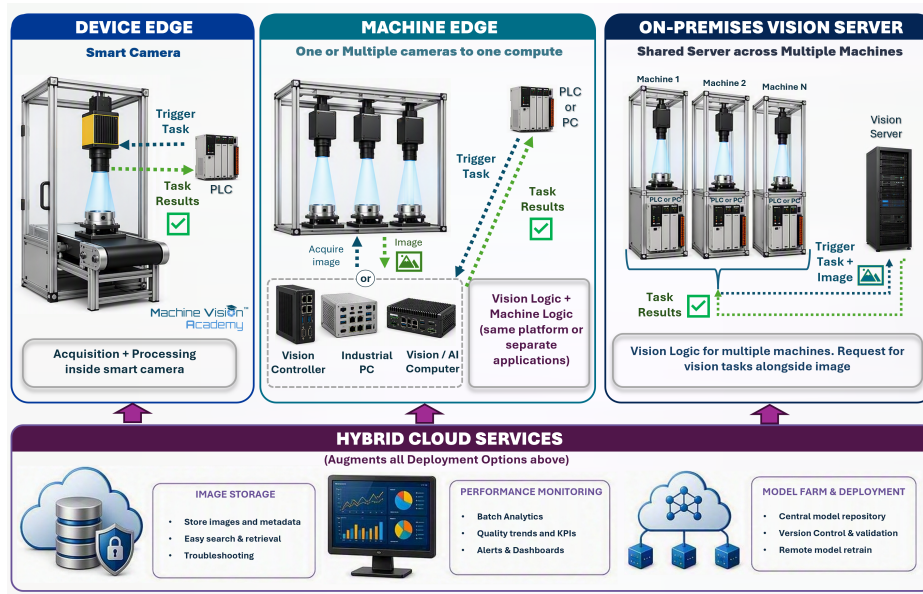


Figure 2.16.: The spectrum of vision architectures: where the processing lives

I/O capable of interacting directly with the machine. The inspection loop can therefore be remarkably short: trigger, acquire, process, decide and communicate - or even switch an output - without an external vision computer in the production path [108], [109].

i Representative Device Edge vision products and families

A non-exhaustive selection of **2D smart-camera families** includes **Cognex** In-Sight, **KEYENCE** VS Series, **Zebra** VS and Iris GTX, **OMRON** FHV7 and MicroHAWK F-Series, **SICK** InspectorP6xx, **Teledyne DALSA** BOA/BOA Pro, **Balluff** BVS SC, **Banner Engineering** VE Series and **wenglor** B60 uniVision [79], [110], [111], [112], [113], [114]. Device Edge also includes specialised **3D smart sensors**, such as **LMI Technologies** Gocator laser profilers (laser triangulation) and snapshot sensors (structured-light 3D) [115], and **Micro-Epsilon** scanCONTROL SMART laser-line profilers [116], where acquisition and at least part of the measurement or inspection processing execute within the sensor

Edge computing is not new to industry. Machine vision applications have used processing close to the process for decades. What has changed is how much capability can now be packaged into the device. An example that illustrates this evolution is the DataMan smart sensors, the first product released by Cognex in 1982, soon after the company was founded. This was the predecessor of the In-sight smart camera and it had one only function, OCR. Contemporary smart cameras combine traditional operations - thresholding, blob analysis, pattern matching, metrology and identification - with increasingly capable deep learning tools.

The advantages follow directly from this integration. With no external computer in the normal inference path, communications latency and network dependencies are reduced, timing can be coupled closely to the machine, and production can continue independently of remote IT or Internet services. Installation is simple and compact: for a suitable application the

vision system may require little more than power, triggering, machine communications and the optical arrangement. A3 similarly characterises smart cameras as self-contained systems particularly well suited to distributed, independent inspection points [109].

Smart cameras are also frequently designed around **configuration or no-code/low-code environments** rather than conventional software programming. This can shorten development for established inspection tasks and makes useful vision functions accessible to a range of engineers who are not software specialists. No-code does not necessarily mean that programming is unavailable: many platforms expose expressions or limited scripting, while products such as Zebra Iris GTX, SICK InspectorP6xx and wenglor B60 provide more extensive customisation routes [111], [113].



The smart-camera paradox

Making a vision application easy to configure does not make machine vision itself easy. No-code environments can produce convincing results very quickly, but they can also hide the assumptions behind optics, image formation, algorithms and validation. From a practitioner perspective, this accessibility is valuable when it reduces implementation effort for a well-understood task; it becomes risky when ease of configuration is mistaken for a substitute for vision knowledge. A robust architecture should be selected because it fits the application, not simply because it is the only environment the developer already knows.

The limitations are largely the consequence of the same integration. Processing power, memory, storage, thermal dissipation, sensor choice and expansion capability are bounded by the camera platform. Scaling often means adding or replacing devices rather than expanding a shared computational resource, and tightly coordinated multi-camera applications can become expensive or awkward. Hardware and software are also commonly tied to one vendor ecosystem. **There are notable exceptions:**

freely programmable platforms such as the Baumer AX series and Vision Components VC Z series retain the Device Edge location while exposing Linux-based development and third-party libraries. Baumer supplies an Ubuntu/Jetson software stack and SDKs, while Vision Components supports embedded Linux and HALCON deployment [117], [118]. These products trade some appliance simplicity for much greater algorithmic freedom.

Device Edge is therefore most attractive when the complete inspection can be contained comfortably within one or a few largely independent devices. The familiar description of a smart camera as a complete vision system in a compact package remains useful; the engineering judgement is to make sure that the application - including plausible future growth - remains inside the performance and software envelope of that package.



Strengths and best fit

- **Short and predictable image-to-decision path** - acquisition, processing and decision remain within the device; particularly valuable where the interval between image capture and machine action is very short.
- **High production autonomy** - little or no dependence on external PCs, servers or IT infrastructure during normal operation.
- **Simple and compact architecture** - fewer components, interfaces and software dependencies to install, diagnose and maintain.
- **Rapid development for established tasks** - integrated no-code/low-code environments can substantially reduce implementation effort.
- **Good failure isolation** - a device failure normally affects one

inspection point rather than a whole group of machines

- **Straightforward lifecycle support** - hardware, runtime and configuration can often be managed and replaced as one vendor-supported device

Best fit: Self-contained applications using one or a small number of **independent cameras**, where the processing fits comfortably within the device and priorities include low latency, local autonomy, compact installation and straightforward maintenance. Typical examples include presence/completeness checks, code reading, straightforward gauging, localised defect inspection and simple robot guidance.

Watchouts

- **Finite computational envelope** - processing performance, memory, storage and thermal capacity are bounded by the selected device.
- **Limited scalability** - increasing processing capacity commonly means adding or replacing devices; this matters where an initially simple application is expected to grow substantially.
- **Multi-camera coordination can become cumbersome** - especially when several views must be synchronised or processed jointly
- **Vendor dependence is usually higher** - conventional smart cameras couple sensor, processor, runtime and development environment, although open programmable platforms provide exceptions
- **No-code can constrain complex applications** - bespoke algorithms, complex sequencing and data handling can become difficult to implement and maintain.
- **Ease of configuration can conceal engineering difficulty** - no-code reduces implementation effort, not the need to under-

stand lighting, optics, variation and validation.

Rule of thumb:

Choose Device Edge because the application benefits from being **self-contained**, not simply because a smart camera is the easiest architecture to programme. If scope or complexity is likely to grow, consider the future software and performance requirements before committing to a tightly integrated platform.

2.4.2. Machine Edge Computing

Machine Edge Computing separates image acquisition from the principal processing hardware while keeping that processing within, beside or directly associated with the production machine. Cameras acquire the images, but the inspection executes on a separate local platform. Compared with Device Edge, this introduces additional components and interfaces; in return it provides much greater freedom in computational resources, camera selection, software architecture and machine integration while retaining most of the latency, availability and data-locality benefits of local processing [108].

Three useful variations can be distinguished within this architecture.

The first is the **dedicated vision controller**. Cameras connect to a purpose-built controller supplied as part of an integrated vision ecosystem. Current examples include the **KEYENCE CV-X**, **OMRON FH**, **Cognex In-Sight 6900** and **Zebra 4Sight** controller families [119], [120], [121], [122]. These products retain much of the appliance-like character of a smart camera - coordinated cameras, I/O, industrial communications and graphical vision software - while supporting more cameras, greater processing capacity or broader acquisition options. Dedicated controllers are often a useful middle ground between an all-in-one camera and a fully bespoke IPC.

The second variation is the **PC-based vision system**, in which one or more industrial cameras connect to a general-purpose industrial PC or workstation. This is usually the most flexible Machine Edge form because the camera technology, acquisition interface, processor, GPU and software environment can be selected relatively independently.



Representative PC-based Machine Edge vision software

A non-exhaustive selection of commercial PC-based machine-vision software includes **Cognex** VisionPro; **Zebra** Aurora Design Assistant and Aurora Imaging Library; **MVTec** HALCON and MERLIC; **NI** LabVIEW with the Vision Development Module; and **Teledyne DALSA** Sherlock. These range from graphical/no-code environments to complete programmable libraries, but all support PC-based Machine Edge deployment [123], [124], [125], [126]. Some environments can also target **Device Edge or embedded hardware**, which can delay part of the deployment decision while the application is being developed. **Zebra** Aurora Design Assistant can deploy to Iris GTX smart cameras; **MVTec** supports HALCON/MERLIC on selected embedded platforms; and **Teledyne DALSA** has offered Sherlock on BOA smart-camera systems [111], [114], [123], [124]. This portability is product-specific: **Cognex** VisionPro applications, for example, are not simply deployed as In-Sight jobs. A less common variation is **NI** LabVIEW with the Vision Development Module, where supported applications can target PCs and real-time controllers and a subset of image-processing functions can execute on FPGA hardware [125]. Open-source alternatives include **OpenCV** for classical image processing and computer vision, and **TensorFlow** and **PyTorch** for learned vision.

The use of deep learning can itself influence architecture selection when model size, memory, acceleration or lifecycle requirements exceed the practical envelope of a smart camera. Once a sufficiently capable Machine Edge platform has been selected, however, classical and learned vision do

not normally need separate computing engines. The same IPC can combine calibrated metrology, conventional pattern matching, 3D processing and GPU-accelerated neural-network inference, allowing each part of the inspection to use the method best suited to it.

The third variation uses a **specialised vision or AI computer**. Compact platforms based on NVIDIA Jetson and similar accelerators combine CPU and GPU/NPU computing in hardware designed for edge deployment. Architecturally this remains Machine Edge: a Jetson running TensorRT and an x86 IPC containing a discrete GPU occupy the same deployment location. They differ in processor architecture, power and cooling, expansion, serviceability and software ecosystem rather than in where the production decision is made.

Machine Edge consequently offers a particularly broad range of engineering choices. Processing can range from a modest fanless controller to a high-performance multicore computer with discrete GPUs. Cameras, optics and lighting can usually be selected independently, using standard interfaces such as GigE Vision and USB3 Vision or dedicated acquisition hardware for CoaXPress and Camera Link HS. This flexibility is well suited to multi-camera systems, high-resolution imaging, computationally demanding 3D or deep-learning applications and projects requiring close integration with PLCs, robots, databases and other software.

Development flexibility is similarly broad. At one end are graphical and no-code environments; at the other are libraries and application frameworks supporting C++, C#, .NET, Python and other languages. This becomes especially valuable when vision is only one part of a larger machine application. A semi-automated process might ask an operator to select a region on an HMI, acquire an image, review an intermediate result, move a mechanism and then continue the vision sequence. Smart-camera logic can implement some of these interactions, but a PC-based architecture generally provides much greater freedom when vision, operator interaction, machine sequencing and data management must behave as one coordinated application.

The price of that freedom is greater system responsibility. The designer may need to manage operating systems, drivers, camera interfaces, GPU runtimes, application software, cybersecurity and updates, and must keep the complete combination supportable throughout the machine lifetime. More components create more interfaces and potential failure points than an appliance-style smart camera, although they also allow cameras, frame grabbers, computers and accelerators to be replaced or upgraded independently. Development effort can therefore range from a short controller configuration exercise to a substantial software-engineering project.

For demanding manufacturing applications this trade-off is often worthwhile. Machine Edge preserves local production autonomy while providing considerably greater computational, software and integration flexibility than Device Edge. In practice it becomes the natural architecture when vision is no longer an isolated inspection but part of a more complex machine-level application.

Strengths and best fit

- **High computational flexibility** - processing can range from a dedicated controller or fanless IPC to multicore CPUs, GPUs and specialised AI accelerators
- **Deep machine-software integration** - vision can form part of a wider application incorporating machine sequencing, operator interaction, robotics, databases and other functions.
- **Broad hardware choice** - cameras, sensors, acquisition interfaces and processing hardware can usually be selected independently.
- **Excellent multi-camera capability** - several cameras can share processing resources and their acquisitions can be synchronised or analysed jointly.

- **Local production autonomy** - time-critical inference remains at the machine without depending on remote servers or cloud connectivity.
- **Expandable architecture** - memory, storage, processing hardware and accelerators can often be upgraded without replacing the complete imaging system.

Best fit: Applications requiring **multiple coordinated cameras, substantial integration with the wider machine application, or processing beyond the practical envelope of Device Edge**. Typical examples include multi-view assembly inspection, high-resolution inspection, 3D bin picking, advanced metrology, demanding deep-learning inference and semi-automated workflows combining vision with operator interaction.

Watchouts

- **Greater integration responsibility** - operating systems, drivers, acquisition hardware, GPU runtimes, application software and industrial communications may all become part of the design.
- **Larger software lifecycle** - cybersecurity, updates and dependency compatibility become part of maintaining the production machine.
- **Slower initial deployment** - additional hardware and software elements must be selected, integrated, commissioned and supported.
- **Complexity can grow quickly** - an open software environment enables sophisticated applications but can also produce software that is difficult to validate and maintain if architecture

is not controlled.

- **A larger failure domain** - where one computer serves several cameras or inspection functions, loss of that computer may disable the complete machine-vision application.

Rule of thumb:

Choose Machine Edge when the application is already moderately or highly complex, **future requirements are uncertain**, or significant growth is expected. It is particularly appropriate for multi-camera coordination, demanding 3D or deep-learning processing, or close integration of vision with the machine software and operator workflow.

2.4.3. On-Premises Vision Server

An **On-Premises Vision Server** moves the principal processing away from the individual machine while retaining it within the factory or organisational computing infrastructure. Cameras or local acquisition systems transmit images to a server, workstation or compute cluster that provides processing services to one or more machines, cells or lines.

This architecture should not be defined simply by the physical size of the computer. The useful distinction is **centralisation and service scope**. A large GPU workstation dedicated to one inspection machine is still Machine Edge even if it sits outside the machine cabinet. An on-premises server, by contrast, is normally managed as shared infrastructure and provides compute, storage, model serving or other software services to multiple production processes.

A common reason to centralise is simply that the required compute no longer belongs comfortably on the factory floor. Large deep-learning or 3D workloads may need high-power GPUs, substantial memory and storage, and cooling arrangements that are awkward to package beside a machine. In dusty, hot, wet or space-constrained production areas, accommodating

such hardware can require a large protected or cooled enclosure. If the process can tolerate a networked inference path, moving this hardware into a clean, controlled server room can be more practical than forcing data-centre-class compute into the machine.

The second attraction is **resource sharing**. Several machines may call different models on the same GPU infrastructure or use multiple instances of a common model. NVIDIA Triton Inference Server is designed for this model-serving pattern, supporting model repositories and concurrent model instances across CPU and GPU resources [127]. Pooling can improve utilisation and lets compute capacity be expanded independently of each production asset rather than sizing every station for its own peak requirement.

Centralisation also makes **lifecycle management and monitoring** easier. Model versions, software libraries, images, inspection results, logs and performance metrics can be managed from common infrastructure. This is useful even where no regulatory requirement applies, and it becomes more important where AI governance is formalised. Under the EU AI Act, providers of **high-risk AI systems** must establish post-market monitoring that systematically collects and analyses performance information throughout the system lifetime [128]. Centralised inference is not required to meet that obligation - edge systems can report telemetry to separate monitoring services - but server architectures naturally support enterprise monitoring, audit trails, version control and controlled deployment of model updates.

The trade-off is that the production network becomes part of the inspection path. Images must reach the server and results return within the available decision time. High-resolution or high-frame-rate systems can generate considerable network traffic, and latency is now influenced by transport, switching, queuing and shared server load as well as image acquisition and inference. Where very short or tightly bounded response is required, keeping the time-critical decision at Device or Machine Edge may remain preferable even if storage, monitoring or model management is centralised.

Centralisation also enlarges the failure domain. Failure of a Machine Edge computer normally affects one machine; failure of a shared server or network component may affect every station that depends on it. As more production relies on shared infrastructure, redundant networking, compute, storage and failover become progressively more valuable. Troubleshooting likewise crosses the OT/IT boundary and can require expertise in vision, networking, servers, cybersecurity and production automation.

Security and data sovereignty remain easier to control than with remote cloud inference because images, models and production data can stay inside the organisation. The architecture is therefore most compelling when compute is impractical to house at the machine, resources can be shared effectively, or centralised model management and monitoring provide enough value to justify the network dependency.

Strengths and best fit

- **High-performance compute away from the machine** - large GPUs, memory and storage can be installed where power, cooling and physical access are easier to engineer
- **Better suited to harsh production environments** - sensitive high-power computing can remain in a clean, controlled room rather than a large protected machine cabinet.
- **Shared computational resources** - expensive CPU/GPU capacity can serve several machines rather than being duplicated at every station.
- **High computational scalability** - server hardware and accelerator capacity can be expanded independently of individual machines.
- **Centralised software, model and monitoring services** - repositories, versions, logs, metrics and image archives can be consolidated.

- **Strong data locality compared with cloud deployment** - production images, intellectual property and models remain within organisational infrastructure.

Best fit: Applications requiring **compute that is impractical to install at the machine**, or where several machines can benefit from shared GPU resources, central model services, large image repositories or enterprise-level monitoring. It is particularly attractive for demanding deep-learning workloads where local-network latency is acceptable.

Watchouts

- **The production network enters the inspection loop** - bandwidth, latency, congestion and network availability can directly affect inspection performance.
- **Timing becomes less predictable** - response includes image transmission, server queuing, workload contention and result communication as well as inference.
- **The failure domain becomes larger** - a server, switch or network failure can affect multiple machines or production lines.
- **Redundancy may become necessary** - production-critical shared infrastructure may require redundant networking, compute, storage and failover.
- **Raw imaging can consume enormous bandwidth** - central processing may be inefficient if many stations continuously transmit large image streams.
- **OT and IT responsibilities converge** - commissioning and support can require machine-vision, network, server and cyber-security expertise.
- **Resource sharing requires capacity engineering** - efficient average utilisation does not guarantee acceptable response

during simultaneous peaks.

Rule of thumb:

Choose an On-Premises Vision Server when **compute requirements, physical/environmental constraints, resource sharing or centralised monitoring justify moving processing away from the machine**, and the application can tolerate the resulting local-network dependency. As centralisation grows, redundancy and capacity planning should grow with it.

2.4.4. Cloud Inference

Cloud Inference places the production vision processing outside the local manufacturing infrastructure. Images or other vision data are transmitted over a wide-area network to a remotely hosted service, where the model or application executes and returns a prediction or decision. This must be distinguished from **cloud-based development, model management or monitoring**, all of which can use cloud infrastructure while production inference remains at Device Edge, Machine Edge or on an On-Premises Vision Server.

There is no shortage of technically capable hosted inference services. **Amazon SageMaker AI** provides managed real-time endpoints with monitoring and automatic scaling [129]; **Microsoft Azure Machine Learning** offers managed online endpoints and customer-managed Kubernetes endpoints [130]; **Google Vertex AI** provides online prediction endpoints for deployed models [131]; and **LandingAI's LandingLens** offers hosted inference as well as Docker and industrial-edge deployment routes [132]. LandingAI was founded by Andrew Ng in 2017, which helped make its data-centric vision workflow particularly visible within industrial AI [133].

The important point is that these services solve a somewhat different scaling

problem from a typical production machine. A web or enterprise service may receive one request now and thousands a few seconds later, so **elasticity and request concurrency** are central design requirements. AWS explicitly positions serverless inference for intermittent or unpredictable traffic [134], while LandingLens describes its hosted option as appropriate for variable inference loads and labels the cloud route as higher latency than its local Docker and LandingEdge options [132]. A manufacturing station is usually the opposite: the number of cameras, approximate image rate and peak workload are known, but a result may be required repeatedly within a defined process window. For that workload, predictability can matter more than elasticity.

This difference explains why Cloud Inference remains unusual for tightly coupled factory-floor inspection. Wide-area network latency and variability, service queues, external connectivity and service availability become part of the inspection loop. A cloud provider can legitimately describe an endpoint as real-time because it answers synchronous low-latency API requests; that does not imply the deterministic timing normally expected of a production control path. Continuous transfer of high-resolution images can also create significant bandwidth costs and extends the cybersecurity and data-governance boundary to production imagery that may reveal product or process intellectual property.

None of this makes cloud inference impossible. It can be sensible where timing is relaxed, request volume is intermittent or highly variable, the image payload is modest, or a remote service provides a capability that would be uneconomic to maintain locally. The practical warning is simply not to confuse **technical possibility with architectural fit**. A cloud endpoint designed to serve a changing population of users is solving a different optimisation problem from a camera inspecting the same product every 300 ms for ten years.

In industrial vision, cloud platforms are often more compelling in **hybrid architectures**. AWS documents manufacturing patterns in which SageMaker is used to build and train models in the cloud while AWS IoT

Greengrass performs inference locally for latency-sensitive inspection [135], [136]. LandingLens similarly separates hosted inference from self-hosted Docker and LandingEdge, the latter connecting directly to industrial cameras and PLCs [132]. Cognex OneVision provides another contemporary example: AI models are developed, managed and governed using cloud services while runtime inspection executes on supported In-Sight systems at the edge [137]. In these arrangements, loss of cloud connectivity may affect monitoring, synchronisation or model distribution without necessarily stopping the immediate production decision.

It is also useful to distinguish **development platforms from serving platforms**. Tools such as Voxel51 FiftyOne are valuable for dataset curation, applying models to datasets and model evaluation, but are better viewed here as development/evaluation infrastructure rather than as a managed factory-floor inference endpoint [138]. Likewise, a product carrying a cloud label may use cloud licensing, collaboration or training while its runtime remains local. The architecture should therefore always be classified by the location of the production decision, not by the marketing label attached to the development environment.

Cloud development can be especially attractive for learned vision because training and experimentation often need substantial GPU resources only intermittently, whereas production inference may need smaller, predictable resources continuously. The development and deployment architectures therefore do not need to match: datasets can be annotated and models trained centrally, then the validated model can be deployed to an IPC, embedded AI computer, supported smart camera or on-premises server. This separation preserves local production autonomy while still exploiting shared development infrastructure.

i Cloud architectures in industrial machine vision

- **Cloud Inference - technically capable, but still unusual for tightly coupled factory-floor control.** Hosted end-

points are attractive for elastic or variable request loads, but WAN dependency, timing variability, image bandwidth and governance make them a poor default for deterministic production inspection.

- **Hybrid Deployment - local inference with cloud services.** The time-critical decision remains at Device/Machine Edge or on-premises, while cloud services provide storage, monitoring, dataset/model management, collaboration and fleet-level visibility.
- **Cloud Development - develop centrally, deploy locally.** Annotation, training and experimentation can use scalable cloud resources while the resulting model is deployed to the production architecture that best meets the machine requirements.

A cloud-based machine-vision platform does not imply that production inference should execute in the cloud. First decide where the image-to-decision path needs to run; then decide which development and supporting functions benefit from cloud infrastructure.

2.4.5. Comparison and selection criteria

When selecting an architecture, a useful starting question is: **where should the production image-to-decision process actually run?** Once that is understood, development tools can be selected to support the deployment rather than allowing familiarity with a particular software package to decide the architecture by accident. It is a common project trap: a convenient prototype becomes the production platform before anyone has asked whether it is the best place for the system to live.

There is rarely one objectively correct answer. This section focuses on technical criteria that can be compared reasonably generally - timing,

processing requirements, camera topology, integration, reliability, maintainability and growth. Real projects are also shaped by criteria that are organisation- and site-specific: approved vendors, internal skills, procurement, spare-parts strategy, legacy systems, cybersecurity rules, support agreements and corporate standardisation. Those constraints can be every bit as decisive as benchmark performance.

 Use this guidance as a decision aid

The comparisons below are intended to support an engineering decision, **not prescribe one**. Similar applications can reasonably use different architectures because the surrounding organisation, site and lifecycle constraints differ. Technology also moves quickly; the useful objective is not to select the architecture that looks most advanced, but the one whose limitations are easiest to live with for the intended production life.

The following table summarises the main technical characteristics of the three architectures normally considered for **on-site production inference**. Cloud Inference is omitted from the table because it remains an exception rather than the default for tightly coupled factory-floor decisions; where it is genuinely appropriate, its WAN, service and governance constraints should be assessed explicitly rather than treated as another local compute option.

 Traffic-light guide for Table 1.2

favourable · neutral · unfavorable

Selection criterion	Device Edge	Machine Edge	On-Premises Vision Server
Image-to-decision latency	Very low	Low	Network- dependent
Timing predictability	Very high	High	Depends on network and shared workload
Processing capacity	Device-bounded	High to very high	Very high
Compute scalability	Limited; add/replace devices	Good; upgrade local hardware	Very good; expand shared resources
Complex application logic	Limited to platform	Good to very high	Good for services; less suited to tight HMI sequencing
Camera flexibility	Integrated/sup- ported sensors	High	High, subject to acquisition architecture
Multi-camera coordination	Limited	Very good	Good where networked acquisition is practical
Hardware / accelerator flexibility	Low	High	Very high
Software / custom-code flexibility	Low to medium	High to very high	High
Machine- controller integration	Very direct	Very good	Indirect / networked

Selection criterion	Device Edge	Machine Edge	On-Premises Vision Server
Physical footprint at machine	Very small	Moderate	Small locally; infrastructure elsewhere
Production autonomy	Very high	Very high	Depends on local network/server infrastructure
Network dependency for inference	None or minimal	None or minimal	Local production network
Failure isolation	Usually one inspection point	Usually one machine/cell	Shared infrastructure can affect several machines
Data locality	Very high	Very high	Within organisational infrastructure
Implementation complexity	Low	Moderate to high	High
Software / infrastructure lifecycle burden	Low; appliance-like	Moderate to high	High; combined IT/OT infrastructure
Expansion / future growth	Limited	Good	Very good
Centralised resource sharing	None	Limited	Very good

Selection criterion	Device Edge	Machine Edge	On-Premises Vision Server
Enterprise monitoring / model governance	Limited unless hybrid	Good with added services	Very good

Table 1.2 - Inference architecture selection summary. Qualitative ratings are comparative guidance rather than absolute performance guarantees.

The table is useful for comparison, but architecture selection often becomes clearer when approached from the opposite direction: **what characteristics of the application should make a particular architecture attractive?**

 When to consider Device Edge

- **The task is bounded and relatively simple.** The complete inspection sits comfortably inside the device software and compute envelope.
- **A very short and predictable image-to-action path matters.** Acquisition, processing and potentially digital I/O can remain within one device.
- **Only one or a few largely independent cameras are required.** Each inspection point can operate autonomously.
- **Compactness and simplicity are priorities.** Reducing computers, operating systems and interfaces has real lifecycle value.

Think twice when: the application could grow substantially, requires bespoke software or tight multi-camera coordination, or coding sequences and tools start becoming complex and hard to troubleshoot.

💡 When to consider Machine Edge

- **The vision application is complex.** The project benefits from independently selecting camera, acquisition, compute and software.
- **Requirements are uncertain or likely to grow.** An IPC/controller preserves significantly more processing, memory, storage and software headroom.
- **Several cameras must act as one system.** Multi-view inspection and synchronised acquisition are easier to coordinate centrally at the machine.
- **Processing is demanding.** High-resolution imaging, 3D, deep learning or mixed classical/learned pipelines justify CPUs, GPUs or other accelerators.
- **Vision must integrate deeply with the wider machine software.** Robotics, databases, HMI interaction and machine sequencing can be coordinated within one application.

Practical warning sign: poorly defined or evolving inspection requirements. Where substantial uncertainty remains, PC-based development - or a software environment that can later target an embedded device - can preserve options while the real requirements become clearer.

💡 When to consider an On-Premises Vision Server

- **The required compute is awkward to install at the machine.** Large GPUs, power and cooling are easier to manage in controlled infrastructure.
- **Several machines can share expensive resources.** Central CPUs, GPUs, memory or storage can serve more than one inspection station.
- **Central model or software services provide value.** Com-

mon models, libraries and versions are easier to manage centrally.

- **Monitoring, storage and governance are important.** Images, results, logs and model performance can be consolidated across machines.
- **The inspection can tolerate a networked processing path.** Network and server response can be kept within the process timing envelope.

Main question: do the benefits of centralisation justify enlarging the failure domain? As more machines depend on the same infrastructure, capacity planning, redundancy and failover become correspondingly more important.

The architecture selected for production does not have to dictate the rest of the vision lifecycle. Local inference can be combined with cloud development, model management, monitoring, image storage and cross-site services. These hybrid arrangements are best considered **after the production inference path has been placed**, rather than treated as another competing location for the time-critical decision.

A practical closing principle is to favour **the simplest architecture that satisfies the production requirements with adequate margin and an acceptable lifecycle burden**. Simplicity does not mean choosing the device that is easiest to configure in a demonstration. It means avoiding compute, software, infrastructure and dependencies that production does not need, while retaining enough headroom for the system to remain maintainable as requirements and technology evolve.

2.5. The vision system as an engineered chain

Previous sections considered what machine-vision systems are asked to do and where their processing can be deployed. This section turns to how the pieces fit together. Whatever the application family or architecture, a working vision system is an engineered chain: each stage creates the conditions required by the next, and weaknesses introduced early in the chain are difficult to recover later. The discussion begins with the stages themselves, then considers the order in which they should be engineered, shows how a requirement propagates through the complete system, and closes with the professional roles, standards and credentials that support the discipline.

2.5.1. The seven stages: presentation, illumination, optics, sensor, processing, decision and action

Machine-vision texts commonly describe a path from image acquisition through image processing to a decision, while industry guidance similarly treats lighting, optics, sensors, processing and communications as interdependent elements of a complete system [95], [139]. For the purposes of this handbook, it is useful to make that chain slightly wider. The five stages that form the core imaging chain - illumination, optics, sensor, processing and decision - are bracketed by two application stages: presentation before the image is acquired, and action after the decision has been made. This gives seven stages in total. The distinction is deliberate: a technically excellent image-processing algorithm is of little value if the part is not presented in a usable way, or if the final result never produces the required machine response.

The five core stages can be stated briefly because each is developed later in the handbook. Illumination controls how the relevant features appear. Optics form and scale the image on the sensor. The sensor converts the optical signal into sampled image data. Processing transforms those pixels

into measurements, features, locations or learned outputs. The decision stage then interprets those outputs against the application requirement: for example pass/fail, a dimensional result, a class, a pose or a confidence value. These are the stages most often associated directly with machine vision, but they do not by themselves make a complete automated application.

The first application bracket is **presentation**. For inspection and measurement, repeatable presentation is usually an advantage: the feature should remain within the usable field of view, focus range and lighting geometry, and acquisition should occur at a controlled point in the process. Guides, nests, rails, fixtures, controlled motion and robot poses are therefore not merely mechanical details; they directly determine how much variation the vision system must tolerate [140]. Triggering and illumination synchronisation form part of the same problem. At production speed, even small timing errors translate into positional variation, which is why industrial cameras commonly provide dedicated trigger and illumination-control interfaces for deterministic acquisition [141].

Presentation also determines how many views of the part must be acquired and how those views are created. If the complete inspection can be performed from one viewpoint, the decision is simple. Where features lie on different faces, at substantially different heights, or around the circumference of a component, the engineer must instead decide whether to **move the part, move the camera, or use multiple cameras**. This choice should be made early because it shifts complexity between the mechanical system and the vision system and can influence camera count, calibration, processing architecture and cycle time.

A useful order of preference is to start with the resources already present in the machine. If a robot or positioning mechanism already handles the component, presenting several faces to a fixed camera may add relatively little hardware and can be an economical solution. Where only a small displacement is required—perhaps between two nearby inspection planes—a simple linear or pneumatic movement of either the part or camera may also be practical. As the required motion becomes larger or more complex,

however, several fixed cameras often become more attractive, particularly when cycle time does not allow sequential movement and acquisition. Multiple cameras increase acquisition, calibration and software complexity, but avoid adding moving mechanisms to the inspection cycle.

The surrounding machine design matters just as much. Moving a part or camera introduces actuators, guarding and safety considerations; moving a camera additionally requires suitable cable routing and management. These additions may be quite reasonable inside an already guarded automated cell, but difficult to justify on an otherwise simple open station. Applications involving calibrated measurement also deserve particular care: a fixed camera provides a stable imaging geometry, whereas moving either the camera or part requires the relevant position and transformation to remain sufficiently repeatable. Robot-mounted cameras are entirely feasible, but they introduce an explicit hand-eye calibration between the camera and robot coordinate systems. Current machine-vision and robotics tools support both fixed-camera and moving-camera configurations, but their calibration becomes part of the engineered system rather than an incidental setup step.

Choosing how to obtain multiple views

If more than one view is required, a useful starting guide is:

- **Move the part** when it is already manipulated by a robot or positioning mechanism and additional poses can be introduced without compromising cycle time.
- **Move the camera** when the part is difficult to manipulate and the required camera movement is simple, repeatable and compatible with calibration, guarding and cable management.
- **Use multiple fixed cameras** when several views are required simultaneously, cycle time is short, movement would be mechanically complex, or maintaining a fixed calibrated geometry is

particularly important.

- **Consider specialised multi-view optics** before adding motion or cameras where the part geometry permits it. Pericentric, catadioptric, hypercentric and multi-mirror optics can capture several surfaces or viewpoints in a single exposure, sometimes replacing a multi-camera or mechanically rotated arrangement.

There is no universal hierarchy: the objective is to place the necessary complexity where it is easiest to control and maintain—mechanics, optics or vision software.

Specialised optics deserve particular mention because they provide a fourth option that is easily overlooked. Opto Engineering, for example, offers **360° and multi-view lenses** designed to capture several surfaces of a component using one camera [142]. Pericentric and catadioptric optics can image the top and lateral surfaces of suitable objects, while multi-mirror designs can place several side views within the same image. For cylindrical parts, caps, bottles, threads and similar geometries, this can remove both mechanical rotation and the need to synchronise several cameras. The trade-off is that these are specialised optical arrangements with constraints on object geometry, working distance, field of view and image utilisation, so they should be considered as another engineering option rather than a universal substitute for multiple cameras.

A second principle of presentation is to **remove as much variation as possible that is not inherent to the product or the inspection requirement**. Uncontrolled changes in position, orientation, background, illumination, surface condition or surrounding equipment increase the range of appearances that the vision system must interpret without adding useful information. This matters for rule-based vision, but it is equally important for deep learning. A model can learn substantial variation when that variation is represented during training—and predictable changes can sometimes be reproduced through augmentation—but unexpected

production conditions introduce *domain shift* and can cause significant performance degradation even in otherwise capable models. In practice, these variations are particularly troublesome because they may appear late in development, during validation or only after deployment, leading to additional data collection, retraining and revalidation. The aim is therefore not to make every image identical, but to ensure that the variation reaching the vision system is intentional, understood and representative of the conditions against which the system has been designed and validated.

The second application bracket is **action**. A verdict that remains inside the camera or PC has not completed the automation task. The result must be delivered to the machine and associated with the correct product at the correct time: a reject mechanism may need to fire, a robot may need a coordinate, a process may need to be adjusted, an image may need to be stored for traceability, or an operator may need to be alerted. Depending on the architecture, this hand-off may use digital I/O, an industrial Ethernet protocol, a robot interface, OPC UA, a database or higher-level manufacturing software. The specific technology matters less than the engineering requirement: the result must reach the right destination, remain associated with the right product, and produce the intended response.

A recurring example from practice illustrates why this wider system boundary matters. Consider an automated inspection station that identifies a defective, non-serialised product but the system is design to rely on an operator to remove it manually once the vision identifies a defect. The vision algorithm may be performing perfectly, yet the original business objective - preventing defective product from continuing through production - still depends on a fallible human action and therefore can fail as an end-to-end solution.

Presentation and action reach into mechanics, robotics, controls, data systems and process engineering, so this handbook does not attempt to cover them with the same depth as illumination, optics, sensors and image processing. They are included in the chain because the vision engineer

must nevertheless understand and influence them. Machine vision is rarely installed in isolation; it succeeds as part of a machine that performs and end-to-end task.

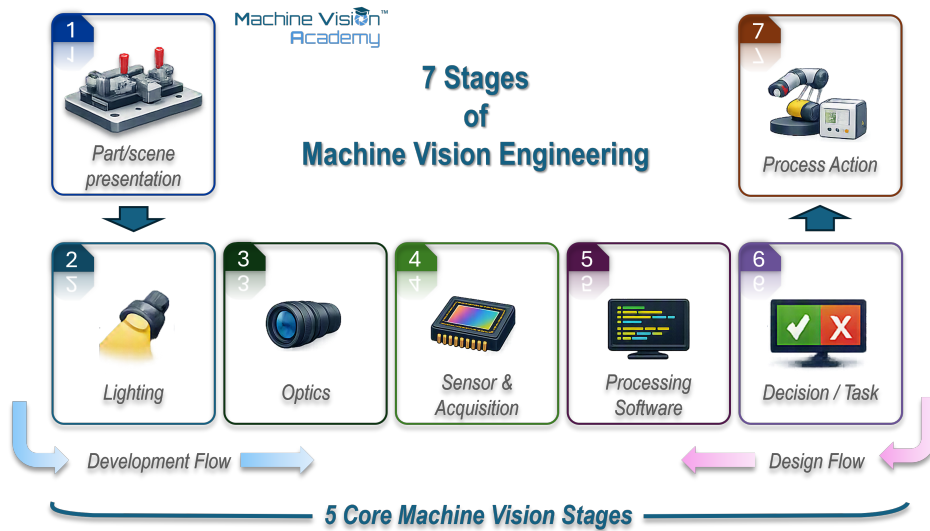


Figure 2.17.: The vision system as a seven-stage engineered chain

The seven stages can also be used as a compact checklist when first framing an application:

Table 2.3.: The seven stages of the engineered vision chain

Stage	What it does	Question to answer	De- veloped in
1 Present- ation	Places the part or scene in a usable state at a known acquisition instant	Is the required feature observable with controlled or intentionally bounded variation?	This section; Ch. 17
2 Illu- mina- tion	Controls how the relevant feature interacts with light	Can the feature be made visible with useful and repeatable contrast?	Chapter 4
3 Op- tics	Forms and scales the image on the sensor	Is the required detail resolved across the necessary field and depth?	Chapter 3
4 Sensor	Converts the optical image into sampled image data	Are resolution, sensitivity, speed, dynamic range and noise suitable?	Chapter 5
5 Pro- cessing	Extracts measurements, features, locations or learned outputs	Can the required information be recovered reliably from the image?	Chapters 6–10
6 De- cision	Maps the processed result to an application outcome	Does the result satisfy the acceptance, classification or localisation requirement?	Ch. 9, 16, 18
7 Ac- tion	Transfers the result and causes the required process response	Does the correct result drive the correct response for the correct product, in time?	This section; Ch. 17

2.5.2. Constraint-first engineering: solve the image before the algorithm

If there is one design principle worth carrying through the rest of this handbook, it is to improve the image in the physical world before compensating for it in software. Image acquisition is not a neutral front end: illumination, surface interaction, optics, focus, exposure, motion and sensor characteristics determine what information is actually recorded [95]. Once a feature has been hidden by glare, clipped by saturation, blurred by motion or never resolved by the optics, downstream processing has far less to work with. Some processing can enhance weak information; it cannot reliably recreate information that the acquisition system never captured.

Lighting normally provides the greatest leverage because the camera observes light from the scene rather than the object in the abstract. Changing the direction, spectrum, diffusion or polarisation of that light can suppress irrelevant structure and make the required feature dominant. Industrial guidance repeatedly makes the same practical point: stable, application-specific illumination simplifies camera configuration and algorithm design, whereas uncontrolled illumination creates variability that must be handled later [143]. Optics and presentation are part of the same argument. A better lens, a more suitable working distance, a polariser, a controlled background or a small change in part angle can turn a difficult image-processing problem into a straightforward one.



The order of attack: physics before software

When a feature is difficult to detect, first ask what can be changed before the image becomes pixels:

- reduce unnecessary variation in part presentation or background;
- change light direction, wavelength, diffusion or polarisation;
- use bright-field, dark-field, backlight, coaxial or another suitable illumination geometry;

- adjust working distance, magnification, aperture, focus or optical filtering;
- shorten exposure and synchronise illumination if motion blur is the limitation;
- only then decide how much algorithmic complexity is genuinely required.

This is not an argument against sophisticated processing. It is an argument for giving that processing the best possible signal.

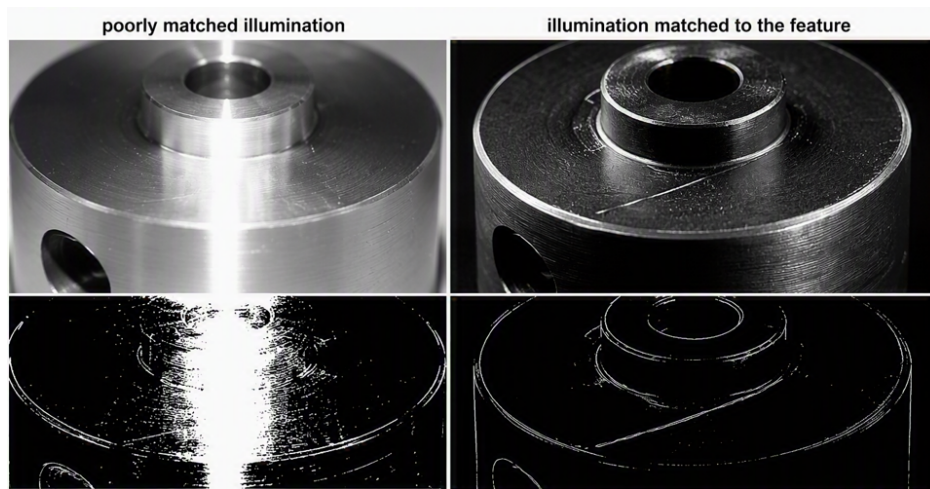


Figure 2.18.: The same feature under poor and well-chosen lighting

Deep learning makes this principle more important, not less. Learned models can tolerate forms of variation that would be awkward to model with hand-designed rules, but they remain dependent on the information and statistical consistency of their input images. Research on computer-vision robustness shows that blur, noise, illumination change and other common image corruptions can materially reduce model performance [144]. Experimental work with controlled active illumination has also shown

that making image appearance more consistent can reduce the amount of training data required to reach a given level of performance [145]. These studies are not industrial quality-inspection recipes, but they reinforce the same engineering message: robustness in the model does not remove the value of robustness in image formation.

The temptation in a deep-learning project is easy to understand. As soon as images can be acquired, it is possible to start collecting data and training models. That can create two expensive traps:

- Poor images become a poor long-term asset. The image dataset is often the most valuable reusable output of a vision application: it can support retraining, root-cause analysis, future inspections and process studies. A well-formed image may contain useful information beyond the original inspection requirement; an image that is blurred, saturated or dominated by uncontrolled reflections has already discarded some of that opportunity.
- Weak image formation can be mistaken for insufficient training. In a rule-based project, a small set of images often reveals quite quickly that a feature is not visible enough. In deep learning, the team may instead respond by collecting more samples, increasing training time, trying larger models or tuning augmentation before questioning the acquisition setup. Because model performance usually changes gradually rather than failing for one obvious physical reason, the point at which the image itself is inadequate can be harder to recognise.
- The model may learn nuisance variation that the machine could have removed. If background, illumination, reflections or pose vary unnecessarily in the training set, a model may devote capacity to those factors instead of the feature that matters. Data augmentation and broader training data are valuable tools, but they should represent variability the deployed system genuinely needs to handle, not compensate by default for an avoidably unstable imaging station.

Deep learning does not repeal image formation

In our experience, one of the easiest mistakes in a deep-learning vision project is to start model development as soon as the first usable images appear. Resist that pressure. Spend enough time proving that the image formation is as strong, stable and informative as it reasonably can be before committing to large-scale data collection. Waiting for a different light, lens, filter or fixture can feel like lost development time. Collecting and labelling thousands of images only to discover later that the feature was poorly illuminated is usually much more expensive.

2.5.3. How the chain interacts: propagate the requirement end to end

The chain is drawn from presentation to action because that is the direction in which information flows during operation. Design, however, starts from the required outcome. A vision engineer should not begin with the question “which camera should I use?” but with a measurable description of what the system must decide or measure, under what production conditions, and with what timing and confidence. From there, each stage constrains the stages around it. This systems view is consistent with the broader computer-vision engineering principle of modelling the image-formation process and testing the complete method under the conditions in which it is expected to operate [146].

Consider a simplified requirement to detect a surface defect approximately 50 micrometres across. That feature must first be represented with enough spatial information and contrast to distinguish it from normal surface variation. The required field of view and sampling then constrain sensor resolution and optical magnification. Magnification, sensor size, working distance and aperture influence lens choice and depth of field; depth-of-field requirements feed back to how tightly the part height or pose must

be controlled. The material and defect geometry constrain illumination because a scratch, dent, contamination mark and colour change interact with light differently. Throughput then constrains exposure time, triggering and processing time. A requirement that initially sounds like a single number therefore becomes a coupled set of optical, mechanical, electrical and computational constraints.

The same coupling appears in less obvious forms. Increasing magnification may improve spatial sampling but reduce field of view and depth of field. Shortening exposure may reduce motion blur but require more illumination or sensor sensitivity. A wider aperture may increase signal but reduce depth of field. Relaxing mechanical fixturing may simplify the machine but increase the search range and algorithmic burden. Moving processing to a shared server may increase compute capacity but add network timing and a larger failure domain. None of these choices is inherently good or bad; the engineering task is to make the trade-offs explicit and keep the whole chain inside the requirement envelope.

This is why feasibility work should try to expose contradictions before hardware and software become fixed. If the required field of view and smallest feature imply an optical setup that cannot maintain focus across the actual part-height variation, the useful result of the feasibility study is not a clever algorithm - it is discovering that the requirement, presentation or imaging geometry must change. The later design framework formalises this process, but the habit starts here: trace important requirements through every stage and ask where they become difficult, sensitive or mutually incompatible.

i Requirements flow backwards; information flows forwards

During operation, the system runs from presentation to action. During design, work backwards from the requirement:

1. What decision or measurement must the system deliver, and what action follows?

2. What image information is necessary to support that result?
3. What illumination, optics and sensor conditions are required to capture that information?
4. What presentation, timing and environmental controls are needed to keep those conditions valid in production?

Then run the reasoning forwards again to verify that the complete image-to-action chain closes within the required cycle time, accuracy and reliability.

2.5.4. Professional roles, standards and certification landscape

Machine vision sits across several engineering disciplines. A single deployment may involve illumination and optics, image sensors, mechanics, electrical design, software, industrial networking, robotics, controls, metrology, data handling and production validation. It is therefore common for different organisations to divide the work differently. In a small team, one engineer may cover most of the chain; in a larger programme, responsibility may be distributed across specialists.

Typical roles around a machine-vision project include:

- Vision or applications engineer - develops the imaging concept and normally owns much of the illumination, optics, acquisition and image-processing chain.
- Systems integrator or machine builder - combines vision with mechanics, controls, safety, motion and the wider machine architecture.
- Controls or automation engineer - integrates triggering, sequencing, communications, interlocks and the resulting machine actions.

- Metrology or quality engineer - defines measurement traceability, acceptance criteria, capability studies and validation where the vision result is used quantitatively.
- Data, AI or software engineer - increasingly involved where deep learning, dataset infrastructure, deployment pipelines or substantial application software form part of the solution.
- End user, process engineer and maintenance team - provide the production knowledge that defines real variation, failure modes, maintainability and long-term ownership.

The boundaries between these roles are deliberately porous. A good vision engineer does not need to be the plant controls expert or the mechanical designer, but must understand enough of those disciplines to recognise when they constrain the image or the final action. Equally, software expertise alone is not a substitute for understanding image formation. That breadth is one reason machine vision is often learned through a mixture of formal study, vendor and association training, laboratory work and production experience.

At the time of editing, one of the most established broad professional credentials in the machine-vision industry is the Association for Advancing Automation (A3) Certified Vision Professional (CVP) programme [147]. CVP-Basic covers the foundations of machine vision through courses in fundamentals, optics, lighting, cameras and image sensors, image processing and vision-system design [148]. CVP-Advanced extends the scope into topics including advanced optics and lighting, 3D vision, machine learning, line-scan imaging, high-speed systems, vision-guided robotics, non-visible imaging, colour, metrology, advanced image processing and system integration [149]. A3 currently offers proctored Basic and Advanced examinations, and states that individual CVP certification is valid for five years [147].

Table 2.4.: Examples of current machine-vision professional certification routes

Pro- gramme	Purpose / scope	What it signifies
A3 CVP- Basic	Broad foundation: machine-vision fundamentals, optics, lighting, cameras/sensors, image processing and system design.	Individual has demonstrated a common baseline of machine-vision knowledge.
A3 CVP- Advanced	Advanced application topics including 3D, machine learning, line scan, high speed, robotics, metrology and integration.	Individual has demonstrated broader and deeper knowledge across advanced vision topics.
EMVA 1288 User / Ex- pert	Specialist knowledge of EMVA 1288 camera/sensor characterisation and use of its measurement data.	Individual has demonstrated knowledge of this specific camera-characterisation standard, not a general machine-vision qualification.

Professional certification should also be distinguished from company-level integration certification. A3 operates a Certified System Integrator programme intended to give end users a benchmark when selecting integration companies; the criteria include company experience and qualified technical personnel rather than certifying a single individual [150]. This distinction is useful for practitioners: an individual credential says something about a person’s demonstrated knowledge, while an integrator credential addresses the capability and track record of an organisation.

In Europe, the European Machine Vision Association (EMVA) has a different but equally important role. It develops and supports industry standards and educational resources rather than offering a direct European

counterpart to the broad CVP qualification. Its EMVA 1288 programme provides User- and Expert-level certification specifically around camera and image-sensor characterisation [151]. The EMVA also hosts GenICam, the generic camera-interface standard used across interfaces such as GigE Vision, USB3 Vision and CoaXPress [152]. At the time of editing, the long-running EMVA 1288 characterisation standard is also being progressed towards the international ISO 24942 standard; ISO lists ISO/DIS 24942 as under development in 2026 [153]. These details will inevitably evolve, so readers should treat the named programmes as examples of the current professional landscape rather than a permanent catalogue.

 Certification is a foundation, not the finish line

A recognised qualification is valuable because it establishes vocabulary, fundamentals and a common technical baseline. It does not replace application judgement.

Machine vision is learned by repeatedly connecting theory to real images, real parts and real production constraints: seeing how a surface changes under a different light, discovering why a nominally correct lens does not provide enough depth of field, or tracing an intermittent false reject back to timing, vibration or presentation. Certification can structure that learning; experience turns it into engineering judgement.

There is also a broader educational challenge. Machine vision is used widely in manufacturing, yet access to structured, vendor-neutral learning remains uneven across universities, companies and regions. One of the longer-term aims of the Machine Vision Academy is therefore to make reusable educational material openly available so that individuals, educators and academic institutions can build courses around a common industrial baseline. At the time of writing this remains an ambition rather than an established accreditation route, but the direction is intentional: make the fundamentals easier to learn, then allow institutions and professional bodies to build recognised learning pathways on top of them. Current project

material is intended to be released through MachineVisionAcademy.org as that programme develops.

The broader message of this section is that machine vision is not a sequence of isolated component choices. It is an engineered chain, owned by people who must reason across the boundaries between image formation, computation and automation. The next section turns that principle into a more explicit method for approaching a new vision problem from the requirement onwards.

2.6. Approaching a vision problem: a design framework

2.6.1. From inspection requirement to imaging specification

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

2.6.2. Characterising the feature: contrast, scale, geometry, tolerance

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

2.6.3. Detection versus measurement: how the task drives every later choice

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

2.6.4. A worked decision path, cross-referenced to later chapters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3. Light, colour and radiometry

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

3.1. Physical foundations of light

3.1.1. Electromagnetic spectrum and the visible band

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.1.2. Radiometric quantities: radiant flux, irradiance, radiance

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.1.3. Photometric quantities and the photopic response

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.1.4. The inverse-square law and cosine law

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.2. Interaction of light with matter

3.2.1. Specular and diffuse reflection; the BRDF

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.2.2. Transmission, refraction and Snell's law

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.2.3. Absorption and fluorescence

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.2.4. Scattering and translucency

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.2.5. Implications for surface inspection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.3. Colour

3.3.1. Trichromacy and the human visual system

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.3.2. Colour matching and the CIE XYZ system

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.3.3. Working colour spaces: RGB, sRGB, HSV, HSL, Lab

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.3.4. Colour temperature and illuminants

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.3.5. White balance and colour constancy

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.4. Polarisation

3.4.1. States of polarisation and Stokes parameters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.4.2. Polarisers, retarders and the Fresnel equations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.4.3. Polarisation-resolving sensors

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

3.4.4. Industrial uses: glare suppression, stress imaging, transparent-material inspection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4. Optics and lens systems

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

4.1. Geometric optics

4.1.1. The pinhole model

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.1.2. Thin- and thick-lens models

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.1.3. Focal length, f-number, aperture

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.1.4. Field of view and working distance

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.1.5. Depth of field and hyperfocal distance

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.2. Real lens behaviour

4.2.1. Spherical and chromatic aberration

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.2.2. Geometric distortion: radial and tangential

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.2.3. Vignetting and cos fall-off

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.2.4. Modulation transfer function and resolving power

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.2.5. Diffraction limit

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.3. Specialty optics

4.3.1. Telecentric lenses for metrology

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.3.2. Macro and micro imaging

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.3.3. Line-scan and Scheimpflug configurations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.3.4. Liquid and variable-focus lenses

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.4. Optical filters

4.4.1. Bandpass, longpass and shortpass filters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.4.2. Polarising filters in imaging

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.4.3. Neutral density, UV cut and IR cut filters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.4.4. Interference filters and dichroic elements

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.4.5. Filter placement and stacking

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.5. Selecting an optical configuration

4.5.1. Matching focal length to sensor size, FOV and working distance

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.5.2. Resolution and DOF budgeting from the feature specification

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.5.3. When telecentricity, filtering or specialty optics are required

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.5.4. Mount standards and mechanical envelope: C, CS, F, M42, S-mount

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

4.5.5. A lens-selection workflow, cross-referenced to Chapters 2 and 5

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

Part II.

Image Formation

i **Part status:** Structural placeholder.
This part introduction will later summarise its scope, prerequisites and learning objectives.

5. Illumination for vision

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

5.1. Illumination principles

5.1.1. Light and geometry as the first solution

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.1.2. Contrast, feature enhancement and defect visibility

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.1.3. The lighting-first design method

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.2. Illumination geometries

5.2.1. Frontal and diffuse illumination

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.2.2. Directional and grazing illumination

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.2.3. Backlighting for silhouettes and dimensioning

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.2.4. Coaxial (on-axis) illumination

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.2.5. Dome and cloudy-day illumination

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.2.6. Dark-field illumination

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.2.7. Structured light

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.3. Light sources

5.3.1. LEDs: colour, drive, lifetime, thermal management

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.3.2. Fluorescent, halogen and xenon sources

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.3.3. Laser sources and safety classes

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.3.4. Continuous, strobed and pulsed operation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.4. Wavelength selection

5.4.1. Visible-band choices and colour separation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.4.2. Near-infrared imaging

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.4.3. Ultraviolet imaging

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.4.4. Short-wave infrared and thermal imaging

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.4.5. Multispectral and hyperspectral approaches

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.5. Environmental integration

5.5.1. Ambient light rejection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.5.2. Enclosures, shrouds and shielding

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.5.3. Vibration, contamination and thermal drift

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.6. Designing an illumination solution

5.6.1. Characterising the feature and the surface

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.6.2. Single-plane versus multi-plane and 3D features

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.6.3. Edge visibility for measurement versus contrast for detection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.6.4. Matching geometry to task: a decision guide

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.6.5. Combining geometry, wavelength and filtering

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

5.6.6. Worked examples: from part to lighting choice

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6. Image sensors and cameras

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

6.1. Sensor technologies

6.1.1. CCD architecture and readout

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.1.2. CMOS architecture: global vs rolling shutter

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.1.3. Line-scan and TDI sensors

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.1.4. SPAD, event-based and neuromorphic sensors

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.1.5. Multispectral and hyperspectral sensor architectures

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.2. Sensor performance and noise

6.2.1. Quantum efficiency and spectral response

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.2.2. Read noise, shot noise, dark current, fixed-pattern noise

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.2.3. Dynamic range and bit depth

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.2.4. Full-well capacity and saturation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.2.5. Signal-to-noise budgeting

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.3. Sampling, resolution and colour acquisition

6.3.1. Pixel pitch, sensor size and effective resolution

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.3.2. Bayer patterns and demosaicing

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.3.3. 3-CCD and multi-sensor colour

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.3.4. Monochrome versus colour trade-offs

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.4. Camera interfaces and image transport

6.4.1. Interface selection criteria: bandwidth, cable length, CPU load, cost

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.4.2. USB3 Vision: bandwidth, cable limits and use cases

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.4.3. Camera Link, CoaXPress and frame-grabber-based interfaces

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.4.4. Legacy and niche interfaces: FireWire, analogue, MIPI CSI

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.4.5. Triggering, exposure control and time stamping

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.5. GigE Vision in depth

6.5.1. Generational bandwidth: 1, 5, 10 and 25 GbE and practical reach

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.5.2. The GigE Vision protocol: discovery, stream and control channels

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.5.3. Network configuration: jumbo packets, dedicated NICs, receive buffers

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.5.4. Multi-camera topologies, switches and bandwidth budgeting

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.5.5. Power over Ethernet and cabling

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.5.6. GenICam and cross-vendor interoperability

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.6. Selecting a camera and interface

6.6.1. From resolution and speed requirement to sensor choice

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.6.2. Matching interface to bandwidth, distance and architecture

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.6.3. Area-scan versus line-scan decision

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

6.6.4. A camera-and-interface selection workflow

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

Part III.

Image Processing

i **Part status:** Structural placeholder.
This part introduction will later summarise its scope, prerequisites and learning objectives.

7. Digital images: representation and pixel-level processing

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

7.1. The digital image as a signal

7.1.1. Sampling, quantisation and the resulting data model

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.1.2. Coordinate systems, indexing and boundary conventions

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.1.3. Colour, greyscale and multi-channel representations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.1.4. File formats and lossless vs lossy compression

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.2. Point operations

7.2.1. Brightness, contrast and gain

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.2.2. Look-up tables, gamma and tone curves

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.2.3. Histogram equalisation and adaptive stretching

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.2.4. Thresholding as a point operation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.3. Arithmetic and logical operations

7.3.1. Pixel-wise arithmetic and blending

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.3.2. Absolute and signed differences

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.3.3. Masking and region-of-interest operations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.3.4. Bitwise operations on binary images

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.4. Noise models and denoising choices

7.4.1. Gaussian, Poisson and impulse noise models

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.4.2. Estimating noise from image statistics

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

7.4.3. Choosing a denoiser: signal preservation vs artefact suppression

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8. Neighbourhood operations and image filtering

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

8.1. Convolution and correlation

8.1.1. Discrete convolution in two dimensions

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.1.2. Separable and rank-one kernels

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.1.3. Boundary handling: zero, reflect, replicate, wrap

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.1.4. Computational cost and efficient implementations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.2. Smoothing filters

8.2.1. Box (mean) filter and integral images

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.2.2. Gaussian filter and selection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.2.3. Median and rank-order filters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.2.4. Bilateral and edge-preserving filters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.2.5. Non-local means

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.3. Derivative and edge filters

8.3.1. First-order operators: Sobel, Prewitt, Scharr

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.3.2. Second-order operators: Laplacian and Laplacian of Gaussian

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.3.3. Gradient magnitude and orientation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.3.4. Sub-pixel edge localisation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.4. The frequency domain

8.4.1. The 2D discrete Fourier transform

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.4.2. Magnitude, phase and their perceptual roles

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.4.3. The convolution theorem

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.4.4. Low-pass, high-pass and band-pass filtering

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.4.5. Frequency-domain inspection of periodic structure

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.4.6. The sampling theorem and aliasing

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.5. Scale space and image pyramids

8.5.1. Scale-space theory in outline

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.5.2. The Gaussian pyramid

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.5.3. The Laplacian pyramid

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.5.4. Multi-resolution processing patterns

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.6. Choosing a filtering strategy

8.6.1. Matching a filter to the noise model and the feature

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.6.2. Preserving edges while suppressing noise

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.6.3. Spatial versus frequency-domain: when each wins

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

8.6.4. Ordering operations in a preprocessing pipeline

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9. Segmentation, morphology and blob analysis

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

9.1. Thresholding methods

9.1.1. Global thresholds and manual selection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.1.2. Otsu's method

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.1.3. Adaptive and locally-adaptive thresholding

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.1.4. Multi-level and hysteresis thresholding

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.2. Region-based segmentation

9.2.1. Region growing and merging

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.2.2. Watershed segmentation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.2.3. Superpixels: SLIC and successors

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.2.4. Graph-cut segmentation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.3. Edge-based segmentation

9.3.1. The Canny detector

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.3.2. Edge linking and boundary tracing

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.3.3. Active contours (snakes) and level sets

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.4. Mathematical morphology

9.4.1. Structuring elements

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.4.2. Erosion and dilation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.4.3. Opening and closing

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.4.4. Hit-or-miss transform and skeletonisation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.4.5. Distance transforms and reconstruction

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.4.6. Greyscale morphology

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.5. Connected-components and blob analysis

9.5.1. Connectivity and labelling algorithms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.5.2. Region properties: area, perimeter, centroid

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.5.3. Geometric moments and Hu invariants

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.5.4. Shape descriptors: circularity, eccentricity, compactness

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.5.5. Filtering blobs by property

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.6. Choosing a segmentation approach

9.6.1. Thresholding versus region versus edge versus learned

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.6.2. Reliability under illumination and contrast variation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

9.6.3. Combining segmentation with morphological clean-up

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10. Feature detection, description and matching

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

10.1. Interest-point detection

10.1.1. The Moravec detector and the corner concept

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.1.2. Harris and Shi–Tomasi corners

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.1.3. Scale-invariant detection: difference of Gaussians

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.1.4. Hessian-based detectors

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.1.5. FAST and AGAST corners for real-time use

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.2. Feature description

10.2.1. SIFT descriptor

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.2.2. SURF descriptor

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.2.3. Binary descriptors: BRIEF, ORB, BRISK, FREAK

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.2.4. Learned descriptors: SuperPoint and successors

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.3. Feature matching

10.3.1. Distance metrics: L2, Hamming

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.3.2. Brute-force and FLANN matching

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.3.3. The Lowe ratio test and cross-check

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.3.4. Robust matching with RANSAC and MSAC

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.4. Template and pattern matching

10.4.1. Sum of squared and absolute differences

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.4.2. Normalised cross-correlation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.4.3. Rotation- and scale-invariant template matching

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.4.4. Geometric pattern matching (edge-model methods)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.5. Line, circle and shape detection

10.5.1. The Hough transform for lines

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.5.2. The Hough transform for circles

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.5.3. The generalised Hough transform

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.5.4. Direct fitting: least-squares lines, circles, ellipses

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.5.5. Robust fitting with RANSAC

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.6. Choosing a feature method

10.6.1. Repeatability, invariance and speed trade-offs

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.6.2. Hand-crafted versus learned features

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

10.6.3. Matching the detector–descriptor pair to the task

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11. Geometric transformations, calibration and registration

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

11.1. Coordinate systems and homogeneous coordinates

11.1.1. Image, camera, world and object frames

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.1.2. Homogeneous representation of points and lines

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.1.3. Matrix representation of transformations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.2. The 2D transformation hierarchy

11.2.1. Rigid, similarity and affine transformations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.2.2. Projective transformations (homographies)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.2.3. Estimation from point correspondences

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.2.4. Robust estimation of homographies

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.3. Image warping and interpolation

11.3.1. Forward vs inverse mapping

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.3.2. Nearest-neighbour, bilinear and bicubic interpolation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.3.3. Anti-aliased warping

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.4. The pinhole camera model

11.4.1. Intrinsic parameters and the camera matrix

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.4.2. Extrinsic parameters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.4.3. Radial and tangential distortion models

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.5. Camera calibration

11.5.1. Zhang's method and the planar-target approach

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.5.2. Calibration targets and image-acquisition strategy

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.5.3. Reprojection error and calibration diagnostics

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.5.4. Uncertainty and re-calibration policy

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.5.5. Hand-eye calibration for robot vision

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.6. Image registration

11.6.1. Feature-based registration pipelines

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.6.2. Intensity-based (direct) registration

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.6.3. Registration for inspection: golden templates

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.6.4. Mosaicing and stitching

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.7. Image-comparison metrics

11.7.1. Absolute and squared differences; PSNR

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.7.2. SSIM and multi-scale SSIM

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.7.3. Histogram-distance metrics

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

11.7.4. Learned perceptual metrics (outline)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

Part IV.

3D Vision and Motion

i **Part status:** Structural placeholder.
This part introduction will later summarise its scope, prerequisites and learning objectives.

12. Stereo and 3D vision

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

12.1. Two-view geometry

12.1.1. The epipolar constraint

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.1.2. The essential and fundamental matrices

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.1.3. Estimation from correspondences

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.1.4. Stereo rectification

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.2. Stereo matching

12.2.1. Block-matching costs: SAD, SSD, NCC, census

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.2.2. Aggregation and disparity selection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.2.3. Semi-global matching

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.2.4. Learned stereo matchers (outline)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.2.5. Disparity to depth and triangulation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.3. Uncertainty in 3D measurement

12.3.1. Depth-error propagation from disparity

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.3.2. Task uncertainty versus datasheet accuracy

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.3.3. Uncertainty budgets for 3D measurement

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.4. 3D imaging modalities

12.4.1. Passive stereo

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.4.2. Structured-light triangulation and fringe projection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.4.3. Time-of-flight cameras

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.4.4. LiDAR and long-range sensing

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.4.5. Photogrammetry, structure from motion and MVS

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.4.6. Comparison and selection guide

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.5. Point clouds and 3D representations

12.5.1. Point-cloud data structures and file formats

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.5.2. Filtering, downsampling and normal estimation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.5.3. Meshing and surface reconstruction

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.5.4. Voxelisation and occupancy representations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.6. Point-cloud registration and alignment

12.6.1. Iterative closest point (ICP) and variants

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.6.2. Feature-based 3D registration

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.6.3. Global alignment strategies

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.7. 3D measurement in practice

12.7.1. Sensor acceptance testing with calibrated artefacts

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.7.2. Surface interaction: specular and translucent materials

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

12.7.3. Warm-up drift and environmental control

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13. Motion, tracking and event-based vision

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

13.1. Motion representation

13.1.1. The brightness-constancy assumption

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.1.2. Sparse vs dense motion fields

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.2. Optical flow

13.2.1. The Lucas–Kanade method

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.2.2. The Horn–Schunck method

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.2.3. Dense flow: Farnebäck and pyramidal schemes

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.2.4. Learned optical flow (outline)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.3. Background modelling and change detection

13.3.1. Frame differencing

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.3.2. Gaussian mixture models

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.3.3. Robust background subtraction

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.4. Object tracking

13.4.1. Template and feature tracking

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.4.2. Mean-shift and correlation-filter trackers

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.4.3. Kalman filters and particle filters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.4.4. Multi-object tracking and data association

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.5. Event-based vision

13.5.1. The event-camera imaging principle

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.5.2. Representations: event frames, time surfaces, voxel grids

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.5.3. Algorithms on event streams

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

13.5.4. Industrial fit and integration considerations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

Part V.

Deep Learning

i **Part status:** Structural placeholder.
This part introduction will later summarise its scope, prerequisites and learning objectives.

14. Machine learning foundations for vision

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

14.1. The learning paradigm

14.1.1. Supervised, unsupervised and self-supervised learning

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.1.2. Training, validation and test sets

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.1.3. Bias, variance and generalisation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.1.4. Cross-validation and model selection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.2. Classical classifiers

14.2.1. Linear and logistic regression

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.2.2. K-nearest neighbours

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.2.3. Support vector machines and kernels

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.2.4. Decision trees, random forests and boosting

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.2.5. Bayesian classifiers

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.3. Feature engineering and dimensionality reduction

14.3.1. Hand-crafted image features

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.3.2. Principal component analysis

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.3.3. Linear discriminant analysis

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.3.4. Manifold learning (outline)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.4. Loss functions and optimisation

14.4.1. Squared, absolute and Huber loss

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.4.2. Cross-entropy and log-loss

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.4.3. Gradient descent and its variants

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.4.4. Regularisation: L1, L2, elastic net

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.5. Evaluation of classifiers

14.5.1. The confusion matrix

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.5.2. Accuracy, precision, recall, F-score

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.5.3. ROC and precision-recall curves; AUC

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.5.4. Class imbalance and industrial reality

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.6. Choosing classical versus deep approaches

14.6.1. Data availability and labelling cost

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.6.2. Interpretability and validation burden

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

14.6.3. When hand-crafted features still win

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15. Neural networks and deep learning

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

15.1. Feed-forward networks

15.1.1. Neurons, layers and activation functions

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.1.2. ReLU, sigmoid, tanh, GELU

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.1.3. Softmax outputs and cross-entropy

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.1.4. Universal approximation and the role of depth

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.2. Backpropagation and automatic differentiation

15.2.1. The chain rule on the computational graph

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.2.2. Forward and reverse-mode differentiation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.2.3. Practical implementation in modern frameworks

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.3. Optimisation of deep models

15.3.1. Stochastic gradient descent and mini-batching

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.3.2. Momentum, Nesterov and Adam

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.3.3. Learning-rate schedules and warm-up

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.3.4. Weight decay, dropout, batch and layer normalisation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.3.5. Data augmentation for images

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.3.6. Early stopping and validation monitoring

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.4. Convolutional neural networks

15.4.1. The convolutional layer and parameter sharing

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.4.2. Pooling, stride and receptive field

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.4.3. Padding conventions and output-size arithmetic

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.4.4. Feature hierarchies and interpretability

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.5. Canonical architectures

15.5.1. LeNet-5, AlexNet, VGG

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.5.2. GoogLeNet / Inception

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.5.3. ResNet and residual connections

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.5.4. DenseNet, MobileNet, EfficientNet

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.5.5. Vision transformers and hybrid models

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.6. Training strategy for industrial data

15.6.1. Transfer learning and fine-tuning

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.6.2. Small-data regimes and few-shot learning

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.6.3. Class-imbalance countermeasures

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

15.6.4. Reproducibility: seeds, versioning, provenance

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16. Deep learning for classification, detection, segmentation and anomaly

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

16.1. Image classification

16.1.1. Whole-image classification pipelines

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.1.2. Multi-label classification

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.1.3. Fine-grained classification

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.1.4. Evaluation with imbalanced industrial datasets

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.2. Object detection

16.2.1. Two-stage detectors: R-CNN, Fast R-CNN, Faster R-CNN

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.2.2. One-stage detectors: YOLO family, SSD, RetinaNet

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.2.3. Anchor-free detectors: FCOS, CenterNet

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.2.4. Transformer-based detection: DETR

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.2.5. Evaluation: IoU, mAP, per-class analysis

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.3. Semantic and instance segmentation

16.3.1. Fully convolutional networks

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.3.2. U-Net and encoder–decoder architectures

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.3.3. DeepLab and dilated convolutions

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.3.4. Mask R-CNN for instance segmentation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.3.5. Panoptic segmentation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.4. Anomaly and defect detection

16.4.1. Reconstruction-based methods: autoencoders and GANs

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.4.2. Embedding-based methods: PatchCore, PaDiM, SPADE

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.4.3. One-class classifiers and Deep SVDD

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.4.4. Benchmarks: MVTec AD and industrial extensions

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.4.5. Choosing between anomaly detection and supervised approaches

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.5. Related tasks in industrial vision

16.5.1. Keypoint detection and pose estimation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.5.2. Deep OCR and OCV

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.5.3. Depth estimation from a single image

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

16.5.4. Self-supervised representation learning

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

Part VI.

Deployment and Compliance

i **Part status:** Structural placeholder.
This part introduction will later summarise its scope, prerequisites and learning objectives.

17. From theory to tools: selection, cross-platform mapping and inspection-flow design

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

17.1. The industrial vision software landscape

17.1.1. Configurable tool sets versus code libraries

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.1.2. Open-source libraries: OpenCV, scikit-image, PyTorch/TF ecosystems

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.1.3. Commercial libraries and SDKs: Halcon, VisionPro, Sapura

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.1.4. Smart-sensor and no-code environments: Keyence, Zebra Aurora, In-Sight

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.1.5. Configuring versus programming: trade-offs and consequences

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2. Cross-platform tool mapping

17.2.1. How to read the mapping tables: theory concept to platform tool

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.2. Acquisition and preprocessing across platforms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.3. Edge, line and circle detection: OpenCV / VisionPro / Halcon / In-Sight / Keyence / Zebra Aurora

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.4. Pattern location and alignment across platforms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.5. Measurement and gauging tools across platforms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.6. Defect and surface-inspection tools across platforms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.7. Identification: barcode, Data Matrix, OCR/OCV across platforms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.8. Deep-learning tools across platforms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.2.9. Limits of equivalence: where tools do not map cleanly

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.3. Selecting and prioritising tools

17.3.1. From feature and task to candidate tools

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.3.2. Prioritisation: robustness, speed, determinism, maintainability

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.3.3. Rule-based versus learned tools for a given problem

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.3.4. A tool-selection decision guide, cross-referenced to theory chapters

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.4. Designing an inspection as a sequence of steps

17.4.1. Decomposing an objective into ordered steps

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.4.2. The step pattern: objective, tools, region, output

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.4.3. Locate–mask–measure–decide as a canonical flow

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.4.4. Worked example: scratch inspection decomposed into steps

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.5. Thresholds, checks and pass/fail logic

17.5.1. Per-step checks: presence, contrast, dimension, position limits

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.5.2. Setting limits and thresholds on tool outputs

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.5.3. Building the overall pass/fail decision from step results

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.5.4. Failure attribution: knowing which step failed and why

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.6. Exposing KPIs and data-driven threshold setting

17.6.1. Choosing KPIs and measured outputs per tool and step

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.6.2. Logging tool outputs for offline analysis

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.6.3. Batch statistical analysis to set thresholds and margins

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.6.4. Separating good/bad populations and setting operating points

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.6.5. Linking threshold setting to the validation of Chapter 18

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.7. Software design practice for vision applications

17.7.1. Modularity, reuse and configuration management

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.7.2. Robustness: handling absent, occluded and out-of-spec parts

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.7.3. Defensive checks, error states and safe defaults

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.7.4. Version control, documentation and change control of vision jobs

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

17.7.5. Performance, cycle time and resource budgeting

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18. Systems integration and industrial interfacing

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

18.1. Industrial communication

18.1.1. EtherNet/IP and PROFINET

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.1.2. Modbus TCP and serial protocols

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.1.3. OPC-UA and information modelling

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.1.4. MQTT and vision-to-cloud patterns

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.2. Integration with PLCs

18.2.1. Handshaking and trigger management

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.2.2. Result data structures and tags

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.2.3. Timing budgets and cycle time

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.3. Integration with robots

18.3.1. Coordinate-frame transformations and hand-eye calibration

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.3.2. Fixed and moving cameras

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.3.3. Pick-and-place and vision-guided assembly

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.3.4. Bin picking and 3D-guided robotics

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.4. Line integration

18.4.1. Reject mechanisms and part tracking

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.4.2. Statistical process-control feedback

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.4.3. Traceability and data-logging architecture

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

18.4.4. Human–machine interfaces for vision stations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19. Validation, measurement uncertainty and sampling for inspection

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

19.1. Defining acceptance

19.1.1. False accept and false reject rates

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.1.2. The confusion matrix in industrial terms

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.1.3. Producer and consumer risk

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.1.4. Setting acceptance criteria from requirements

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.2. Repeatability and reproducibility

19.2.1. Sources of variation in a vision measurement

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.2.2. Attribute Gauge R&R: method and calculation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.2.3. Variable Gauge R&R by the ANOVA method

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.2.4. Kappa statistics and inter-rater/inter-system agreement

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.2.5. Interpreting %R&R, %tolerance and number of distinct categories

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.3. Confidence, tolerance and statistical intervals

19.3.1. Confidence intervals for proportions and means

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.3.2. Tolerance intervals and coverage

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.3.3. Confidence–reliability statements (e.g. 95/95)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.3.4. Propagation of uncertainty through a measurement chain

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.4. Sampling plans for inspection validation

19.4.1. Attributes versus variables sampling

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.4.2. Binomial and hypergeometric bases for attribute plans

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.4.3. Operating-characteristic curves, AQL and RQL/LTPD

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.4.4. Zero-acceptance ($c=0$) plans and their use

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.4.5. Confidence–reliability sample-size formulae

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.4.6. ANSI/ASQ Z1.4, Z1.9 and ISO 2859 in practice

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.5. Measurement-system analysis for vision

19.5.1. Bias, linearity and stability studies

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.5.2. Uncertainty budgets and traceability

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.5.3. Total measurement uncertainty for vision gauges

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.6. Verification, troubleshooting and re-validation

19.6.1. A systematic troubleshooting method

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.6.2. Isolating optical, lighting, sensor and software failures

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.6.3. Recording and root-cause analysis

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

19.6.4. Change control and re-validation triggers

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20. Deployment, monitoring and human-in-the-loop for learning-based vision

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

20.1. From trained model to production system

20.1.1. Model export and interchange (ONNX)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.1.2. Quantisation, pruning and operator fusion

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.1.3. Runtime targets: CPU, GPU, edge accelerators

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.1.4. Revalidation of the deployed artefact

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.2. Dataset governance

20.2.1. Collection planning against the operating envelope

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.2.2. Labelling protocols and inter-annotator agreement

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.2.3. Curation, exclusion records and provenance

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.2.4. Versioning and model-to-data traceability

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.3. Post-deployment monitoring

20.3.1. Domain shift and concept drift

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.3.2. Statistical drift detectors

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.3.3. Retraining loops and change control

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.4. Human-in-the-loop systems

20.4.1. Why humans remain in critical and deep-learning loops

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.4.2. Review triggers: low confidence, drift, novelty, borderline cases

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.4.3. Interface and workflow design for human review

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.4.4. Human-in-the-loop as a labelling and improvement engine

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.4.5. Levels of autonomy and human oversight

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.4.6. Automation bias and the limits of human checking

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.5. Explainability and interpretability

20.5.1. Saliency and Grad-CAM

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.5.2. LIME and local surrogates

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.5.3. SHAP and Shapley attributions

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.5.4. Limits of post-hoc explanation

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.5.5. Explanation as debugging versus explanation for audit

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.6. Bias and fairness in industrial vision

20.6.1. Subgroups in industrial data: cavities, suppliers, shifts

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.6.2. Subgroup analysis and reporting

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.6.3. Mitigations: data, loss, imaging

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

20.6.4. Fairness when the system observes people

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21. Process validation and regulation of vision inspections in regulated industries

i **Chapter status:** Structural placeholder.
This chapter currently contains the approved heading structure but no reviewed technical content.

21.1. Process validation of inspection systems

21.1.1. IQ, OQ and PQ for vision systems

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.1.2. Validating an inspection against its acceptance criteria

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.1.3. Linking validation to the sampling and R&R methods of Chapter 18

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.1.4. Software validation and computer-system validation (CSV/CSA)

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.1.5. Validating learning-based inspections: special considerations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.2. The EU Artificial Intelligence Act

21.2.1. Scope, definitions and territorial reach

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.2.2. Prohibited practices

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.2.3. The high-risk category and Annex III

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.2.4. Obligations for providers and deployers of high-risk AI

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.2.5. Limited- and minimal-risk tiers

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.2.6. Timelines, enforcement and penalties

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.3. US regulation for AI and inspection

21.3.1. The FDA framework for AI/ML-enabled devices

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.3.2. Predetermined change-control plans and the total-product-lifecycle approach

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.3.3. 21 CFR Part 11: electronic records and signatures

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.3.4. NIST AI Risk Management Framework

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.3.5. Comparison of EU and US approaches

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.4. Medical-device and pharmaceutical inspection

21.4.1. MDR and IVDR requirements relevant to vision inspection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.4.2. FDA 21 CFR 820 / QSR and process validation expectations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.4.3. GMP and pharma visual-inspection requirements

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.4.4. Validating automated visual inspection against manual inspection

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.5. Other high-risk and safety-critical sectors

21.5.1. Aerospace inspection and qualification expectations

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.5.2. Automotive quality and functional-safety context

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

21.5.3. Common threads across regulated industries

i **Content status:** Placeholder.
This subsection has not yet been drafted or technically reviewed.

Bibliography

- [1] European Machine Vision Association, ‘Machine vision fundamentals: Outline and principles’. in Machine vision fundamentals. European Machine Vision Association, Barcelona, Spain, 2016. Accessed: Aug. 08, 2026. [Online]. Available: <https://www.emva.org/education/machine-vision-fundamentals/>
- [2] VDI/VDE/VDMA, ‘VDI/VDE/VDMA 2632 Blatt 2:2024-11: Machine vision—preparation of a requirement specification and a system specification’, VDI Verein Deutscher Ingenieure e.V., Dusseldorf, Germany, Standard VDI/VDE/VDMA 2632 Blatt 2:2024-11, 2024.
- [3] E. R. Davies, *Computer and machine vision: Theory, algorithms, practicalities*, 4th edn. Waltham, MA, USA: Academic Press/Elsevier, 2012.
- [4] British Machine Vision Association, ‘Home’. BMVA. Accessed: Aug. 08, 2026. [Online]. Available: <https://www.bmva.org/>
- [5] International Organization for Standardization and International Electrotechnical Commission, ‘ISO/IEC 22989:2022: Information technology—artificial intelligence—artificial intelligence concepts and terminology’, International Organization for Standardization/International Electrotechnical Commission, Geneva, Switzerland, International Standard ISO/IEC 22989:2022, 2022.
- [6] Cognex Corp., ‘PatMax overview’. Cognex Vision Library Documentation. Accessed: Aug. 08, 2026. [Online]. Available: https://support.cognex.com/docs/cvl_900/web/EN/cvl_vision_tools/Content/Topics/VisionTools/PatMax_Overview.htm

- [7] Cognex Corp., ‘TrainPatMaxPattern’. In-Sight Spreadsheet Vision Tools Documentation. Accessed: Aug. 08, 2026. [Online]. Available: https://support.cognex.com/docs/is2d_2410/web/EN/InSight_Sheet/Content/Topics/Spreadsheet/VisionTools/TrainPatMaxPattern.htm
- [8] MVTec Software GmbH, ‘Find_shape_model’. HALCON Operator Reference. Accessed: Aug. 08, 2026. [Online]. Available: https://www.mvtec.com/doc/halcon/2105/en/find_shape_model.html
- [9] Advertising Standards Authority, ‘AI as a marketing term report’, Advertising Standards Authority, London, U.K., Nov. 2024. Accessed: Aug. 08, 2026. [Online]. Available: <https://www.asa.org.uk/static/eb77bee0-a147-49b8-81856c3c83586bf2/AI-as-a-Marketing-Term-Report.pdf>
- [10] National Institute of Standards and Technology, ‘First digital image,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.nist.gov/mathematics-statistics/first-digital-image>
- [11] NASA Jet Propulsion Laboratory, ‘Computer process brightens surveyor moon pictures,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.jpl.nasa.gov/news/computer-process-brightens-surveyor-moon-pictures/>
- [12] NASA Jet Propulsion Laboratory, ‘Ranger 7,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.jpl.nasa.gov/missions/ranger-7/>
- [13] L. G. Roberts, ‘Machine perception of three-dimensional solids,’ PhD thesis, Dept. Electrical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA, 1963. Available: <https://dspace.mit.edu/handle/1721.1/11589>
- [14] S. Papert, ‘The summer vision project,’. 1966. Available: <https://people.csail.mit.edu/brooks/idocs/AIM-100.pdf>
- [15] W. S. Boyle and G. E. Smith, ‘Charge coupled semiconductor devices,’ *Bell System Technical Journal*, vol. 49, no. 4, pp. 587–593, 1970, doi: [10.1002/j.1538-7305.1970.tb01790.x](https://doi.org/10.1002/j.1538-7305.1970.tb01790.x).

- [16] J. Goodrich, ‘The first digital camera was the size of a toaster,’ 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://spectrum.ieee.org/first-digital-camera-history>
- [17] M.-K. Hu, ‘Visual pattern recognition by moment invariants,’ *IRE Transactions on Information Theory*, vol. 8, no. 2, pp. 179–187, 1962, doi: [10.1109/TIT.1962.1057692](https://doi.org/10.1109/TIT.1962.1057692).
- [18] A. Rosenfeld and J. L. Pfaltz, ‘Sequential operations in digital picture processing,’ *Journal of the ACM*, vol. 13, no. 4, pp. 471–494, 1966, doi: [10.1145/321356.321357](https://doi.org/10.1145/321356.321357).
- [19] N. Otsu, ‘A threshold selection method from gray-level histograms,’ *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979, doi: [10.1109/TSMC.1979.4310076](https://doi.org/10.1109/TSMC.1979.4310076).
- [20] D. H. Ballard, ‘Generalizing the hough transform to detect arbitrary shapes,’ *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981, doi: [10.1016/0031-3203\(81\)90009-1](https://doi.org/10.1016/0031-3203(81)90009-1).
- [21] M. A. Fischler and R. C. Bolles, ‘Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,’ *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981, doi: [10.1145/358669.358692](https://doi.org/10.1145/358669.358692).
- [22] J. Canny, ‘A computational approach to edge detection,’ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986, doi: [10.1109/TPAMI.1986.4767851](https://doi.org/10.1109/TPAMI.1986.4767851).
- [23] C. Harris and M. Stephens, ‘A combined corner and edge detector,’ in *In proceedings of the 4th alvey vision conference, manchester, u.k.*, 1988, pp. 147–151. doi: [10.5244/C.2.23](https://doi.org/10.5244/C.2.23).
- [24] K. Pearson, ‘LIII. On lines and planes of closest fit to systems of points in space,’ *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901, doi: [10.1080/14786440109462720](https://doi.org/10.1080/14786440109462720).
- [25] M. Turk and A. Pentland, ‘Eigenfaces for recognition,’ *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991, doi: [10.1162/jocn.1991.3.1.71](https://doi.org/10.1162/jocn.1991.3.1.71).

- [26] D. Marr, *Vision: A computational investigation into the human representation and processing of visual information*. 1982.
- [27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, ‘Learning representations by back-propagating errors,’ *Nature*, vol. 323, pp. 533–536, 1986, doi: [10.1038/323533a0](https://doi.org/10.1038/323533a0).
- [28] SRI International, ‘Shakey the robot,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.sri.com/hoi/shakey-the-robot/>
- [29] Stanford University, ‘Stanford’s robotics legacy,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://news.stanford.edu/stories/2019/01/stanfords-robotics-legacy>
- [30] International Federation of Robotics, ‘Robot history - 1983: Flexible automated assembly lines,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://ifr.org/robot-history>
- [31] R. Y. Tsai and R. K. Lenz, ‘A new technique for fully autonomous and efficient 3D robotics hand/eye calibration,’ *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 345–358, 1989, doi: [10.1109/70.34770](https://doi.org/10.1109/70.34770).
- [32] P. J. Besl and N. D. McKay, ‘A method for registration of 3-d shapes,’ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992, doi: [10.1109/34.121791](https://doi.org/10.1109/34.121791).
- [33] S. Hutchinson, G. D. Hager, and P. I. Corke, ‘A tutorial on visual servo control,’ *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996, doi: [10.1109/70.538972](https://doi.org/10.1109/70.538972).
- [34] E. R. Fossum, ‘CMOS image sensors: Electronic camera-on-a-chip,’ *IEEE Transactions on Electron Devices*, vol. 44, no. 10, pp. 1689–1698, 1997, doi: [10.1109/16.628824](https://doi.org/10.1109/16.628824).
- [35] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, ‘Gradient-based learning applied to document recognition,’ *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, doi: [10.1109/5.726791](https://doi.org/10.1109/5.726791).

- [36] P. Viola and M. Jones, ‘Rapid object detection using a boosted cascade of simple features,’ in *In proc. 2001 IEEE computer society conference on computer vision and pattern recognition (CVPR), kauai, HI, USA*, 2001, pp. I-511-I-518. doi: [10.1109/CVPR.2001.990517](https://doi.org/10.1109/CVPR.2001.990517).
- [37] D. G. Lowe, ‘Distinctive image features from scale-invariant keypoints,’ *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004, doi: [10.1023/B:VISI.0000029664.99615.94](https://doi.org/10.1023/B:VISI.0000029664.99615.94).
- [38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, ‘ImageNet: A large-scale hierarchical image database,’ in *In proc. 2009 IEEE conference on computer vision and pattern recognition, miami, FL, USA*, 2009, pp. 248–255. doi: [10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- [39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ‘ImageNet classification with deep convolutional neural networks,’ 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html>
- [40] R. Girshick, J. Donahue, T. Darrell, and J. Malik, ‘Rich feature hierarchies for accurate object detection and semantic segmentation,’ in *In 2014 IEEE conference on computer vision and pattern recognition, columbus, OH, USA*, 2014, pp. 580–587. doi: [10.1109/CVPR.2014.81](https://doi.org/10.1109/CVPR.2014.81).
- [41] O. Ronneberger, P. Fischer, and T. Brox, ‘U-net: Convolutional networks for biomedical image segmentation,’ pp. 234–241, 2015. doi: [10.1007/978-3-319-24574-4_28](https://doi.org/10.1007/978-3-319-24574-4_28).
- [42] K. He, X. Zhang, S. Ren, and J. Sun, ‘Deep residual learning for image recognition,’ in *In 2016 IEEE conference on computer vision and pattern recognition, las vegas, NV, USA*, 2016, pp. 770–778. doi: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [43] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, ‘You only look once: Unified, real-time object detection,’ in *In 2016 IEEE conference on computer vision and pattern recognition, las vegas, NV, USA*, 2016, pp. 779–788. doi: [10.1109/CVPR.2016.91](https://doi.org/10.1109/CVPR.2016.91).

- [44] A. D. et al., ‘An image is worth 16x16 words: Transformers for image recognition at scale,’ in *In international conference on learning representations (ICLR)*, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://openreview.net/forum?id=YicbFdNTTy>
- [45] OpenCV, ‘Anniversary: 25 years of OpenCV,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://opencv.org/anniversary/>
- [46] S. Pichai, ‘TensorFlow: Smarter machine learning, for everyone,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://blog.google/innovation-and-ai/products/tensorflow-smarter-machine-learning-for/>
- [47] M. A. et al., ‘TensorFlow: Large-scale machine learning on heterogeneous systems,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.tensorflow.org/about/bib>
- [48] PyTorch Team, ‘PyTorch, a year in...,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://pytorch.org/blog/a-year-in/>
- [49] PyTorch Team, ‘PyTorch strengthens its governance by joining the linux foundation,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://pytorch.org/blog/PyTorchfoundation/>
- [50] Association for Advancing Automation, ‘A look at smart cameras and vision sensors,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/vision/industry-insights/a-look-at-smart-cameras-and-vision-sensors>
- [51] Automated Imaging Association, ‘AIA announces 2004-2005 events,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/vision/news/aia-announces-2004-2005-events>
- [52] Association for Advancing Automation, ‘The latest advancements in vision standards,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/vision/industry-insights/latest-advancements-vision-standards>
- [53] European Machine Vision Association, ‘EMVA standard 1288 - introduction,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.emva.org/standards-technology/emva-1288/emva-standard-1288/>

- [54] European Machine Vision Association, ‘GenICam history,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.emva.org/standards-technology/genicam/genicam-history/>
- [55] LMI Technologies, ‘LMI technologies launches new gocator all-in-one 3D smart snapshot sensor,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://lmi3d.com/news/lmi-technologies-launches-new-gocator-all-in-one-3d-smart-snapshot-sensor/>
- [56] Cognex Corporation, ‘Cognex introduces world’s first industrial smart camera powered by deep learning,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.cognex.com/en-in/company/press-releases/2020/cognex-introduces-first-industrial-smart-camera-powered-by-deep-learning>
- [57] OMRON Corporation, ‘OMRON releases FH series vision system with industry’s first defect detection AI,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.omron.com/global/en/news/2020/06/c0629.html>
- [58] Micro-Epsilon Messtechnik GmbH & Co. KG, ‘Micro-epsilon history,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.micro-epsilon.com/company/history/>
- [59] KEYENCE Corporation of America, ‘About us - timeline,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.keyence.com/about-us/>
- [60] National Instruments, ‘The measurement revolution - timeline,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.ni.com/nati/annual/00/timeline>
- [61] Matrox Video, ‘History of innovation,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://video.matrox.com/en/about/history-of-innovation>

- [62] Teledyne DALSA, ‘DALSA reinforces leadership in 4K motion picture capture...’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.teledynedalsa.com/en/news/newsroom/dalsa-reinforces-leadership-in-4k-motion-picture-capture-with-introduction-of-new-camera-models-on-board-data-recorder-and-anamorphic-lenses/>
- [63] Cognex Corporation, ‘Cognex history’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.cognex.com/en-cz/company/history>
- [64] Association for Advancing Automation, ‘Robotic vision advances center around software, technology leaps’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/industry-insights/robotic-vision-advances-center-around-software-technology-leaps>
- [65] Basler AG, ‘Milestones’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.baslerweb.com/en-us/company/milestones/>
- [66] National Instruments, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.ni.com/pdf/manuals/322363b.pdf>
- [67] Automated Imaging Association, ‘Smart vision sensors’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/vision/industry-insights/smart-vision-sensors-consultancy>
- [68] Vision Components GmbH, ‘Embedded vision: A success story’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.vision-components.com/en/embedded-vision-a-success-story/>
- [69] MVTec Software GmbH, ‘Press material - company profile’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.mvtec.com/press/press-material>
- [70] ‘Machine-vision system comes compact’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.vision-systems.com/cameras-accessories/article/16739848/machine-vision-system-comes-compact>

- [71] Association for Advancing Automation, ‘Smart cameras: The last step in machine vision evolution?’, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/vision/industry-insights/smart-cameras-the-last-step-in-machine-vision-evolution>
- [72] Cognex Corporation, ‘Cognex enters the machine vision sensor business,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/vision/news/cognex-enters-the-machine-vision-sensor-business>
- [73] Cognex Corporation, ‘Annual report 2004, natick, MA, USA, 2005. [online]’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: https://www.annualreports.com/HostedData/AnnualReportArchive/c/NASDAQ_CGNX_2004.pdf
- [74] Cognex Corporation, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.sec.gov/Archives/edgar/data/851205/000095013502001368/b41828cce10-k.htm>
- [75] Cognex Corporation, ‘Robot communications - kuka robot controller - serial,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: https://docs.cognex.com/is_611/web/EN/ise/Content/Communications_Reference/Kuka_Communications_Serial.htm
- [76] ‘Matrox imaging introduces programmable smart camera,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.edn.com/matrox-imaging-introduces-programmable-smart-camera/>
- [77] ‘Smart cameras get remote heads,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.edn.com/smart-cameras-get-remote-heads/>
- [78] FANUC Robotics, ‘FANUC robotics introduces robot vision system,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automation.com/article/fanuc-robotics-introduces-robot-vision-system>
- [79] KEYENCE Corporation of America, ‘Vision system with built-in AI - VS series,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.keyence.com/products/vision/vision-sys/vs/>

- [80] Teledyne Technologies Incorporated and DALSA Corporation, ‘Teledyne and DALSA complete plan of arrangement,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.teledyne.com/en-us/news/Pages/teledyne-and-dalsa-complete-plan-of-arrangement.aspx>
- [81] Cognex Corporation, ‘Vision guided robotics - KUKA,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.cognex.com/industries/vision-guided-robotics>
- [82] KUKA Roboter GmbH, 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.manualslib.com/manual/1647894/Kuka-Visiontech-2-1.html>
- [83] Cognex Corporation, ‘Cognex acquires maker of deep learning software for industrial machine vision,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://investor.cognex.com/news/news-details/2017/Cognex-Acquires-Maker-of-Deep-Learning-Software-for-Industrial-Machine-Vision/default.aspx>
- [84] OMRON Corporation, ‘OMRON to introduce FHV7-series smart camera with world’s first multi-color light and 12 mpix image sensor,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.omron.com/media/press/2018/11/c1105.html>
- [85] Zebra Technologies Corporation, ‘Zebra technologies completes acquisition of matrox imaging,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://investors.zebra.com/news-and-events/news/news-details/2022/Zebra-Technologies-Completes-Acquisition-of-Matrox-Imaging/default.aspx>
- [86] National Museum of American History, Smithsonian Institution, ‘Supermarket scanner,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: https://americanhistory.si.edu/collections/object/nmah_892778
- [87] DENSO WAVE INCORPORATED, ‘QR code development story,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.denso-wave.com/en/technology/vol1.html>

- [88] University of Cambridge Computer Laboratory, ‘The trojan room coffee pot timeline,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.cl.cam.ac.uk/coffee/qsf/timeline.html>
- [89] P. M. Corcoran, P. Bigioi, E. Steinberg, and A. Pososin, ‘Automated in-camera detection of flash-eye defects,’ *IEEE Transactions on Consumer Electronics*, vol. 51, no. 1, pp. 11–17, 2005, doi: [10.1109/TCE.2005.1405692](https://doi.org/10.1109/TCE.2005.1405692).
- [90] Microsoft, ‘The future of entertainment starts today as kinect for xbox 360 leaps and lands at retailers nationwide,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://news.microsoft.com/source/2010/11/04/the-future-of-entertainment-starts-today-as-kinect-for-xbox-360-leaps-and-lands-at-retailers-nationwide/>
- [91] Guinness World Records, ‘Fastest-selling gaming peripheral,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.guinnessworldrecords.com/world-records/fastest-selling-gaming-peripheral>
- [92] Google, ‘Picture this: A fresh approach to photos,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://blog.google/products-and-platforms/products/photos/picture-this-fresh-approach-to-photos/>
- [93] Apple, ‘The future is here: iPhone x,’. 2026. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.apple.com/uk/newsroom/2017/09/the-future-is-here-iphone-x/>
- [94] M. D. A. et al., ‘Pivotal trial of an autonomous AI-based diagnostic system for detection of diabetic retinopathy in primary care offices,’ *npj Digital Medicine*, vol. 1, 2018, doi: [10.1038/s41746-018-0040-6](https://doi.org/10.1038/s41746-018-0040-6).
- [95] J. Beyerer, F. Puente León, and C. Frese, *Machine vision: Automated visual inspection: Theory, practice and applications*. Springer Berlin Heidelberg, 2016. doi: [10.1007/978-3-662-47794-6](https://doi.org/10.1007/978-3-662-47794-6).

- [96] International Organization for Standardization, *Sampling procedures for inspection by variables—part 1: Specification for single sampling plans indexed by acceptance quality limit (AQL) for lot-by-lot inspection for a single quality characteristic and a single AQL*. International Organization for Standardization, 2022.
- [97] International Organization for Standardization, *Sampling procedures for inspection by attributes—part 1: Sampling schemes indexed by acceptance quality limit (AQL) for lot-by-lot inspection*. International Organization for Standardization, 1999.
- [98] AIMS Industrial, ‘Vision measuring system: CNC video metrology guide’. Accessed: Aug. 09, 2026. [Online]. Available: <https://aimsindustrial.com.au/blogs/product-guides/vision-measuring-system-guide>
- [99] International Organization for Standardization, *Geometrical product specifications (GPS)—acceptance and reverification tests for coordinate measuring machines (CMM)—part 7: CMMs equipped with imaging probing systems*. International Organization for Standardization, 2011.
- [100] Overview AI, ‘Machine vision vs. AI vision for quality control’. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.overview.ai/blog/machine-vision-vs-ai-vision-quality-control/>
- [101] J. Zanon Diaz, M. A. Farooq, and P. Corcoran, ‘Automatic inspection of seal integrity in sterile barrier packaging: A deep learning approach’, *IEEE Access*, vol. 12, pp. 22904–22927, 2024, doi: [10.1109/ACCESS.2023.3348779](https://doi.org/10.1109/ACCESS.2023.3348779).
- [102] J. Zanon Diaz and P. Corcoran, ‘Surface seal image dataset of sterile barrier packaging’, *Data in Brief*, vol. 57, p. 110996, 2024, doi: [10.1016/j.dib.2024.110996](https://doi.org/10.1016/j.dib.2024.110996).
- [103] Cognex Corporation, ‘UDI marking and verification for medical devices’. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.cognex.com/en/applications/barcode-scanning-and-tracking/udi-marking-and-verification-for-medical-devices>

- [104] TechnoMark, ‘What is data matrix ECC200?’ Accessed: Aug. 09, 2026. [Online]. Available: <https://www.technomark-inc.com/cracking-the-code-what-is-datamatrix-ecc200/>
- [105] C. Stokes, ‘Automation technologies drive change in medical device manufacturing’. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.automate.org/industry-insights/medical-device-manufacturing-automation>
- [106] KEYENCE Corporation, ‘Optical character reading (OCR) and 1D/2D code verification’. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.keyence.com/products/vision/applications/ocr-verification-and-character-inspection.jsp>
- [107] KEYENCE Corporation, ‘3D vision-guided robotic bin picking’. Accessed: Aug. 09, 2026. [Online]. Available: <https://www.keyence.com/products/vision/applications/3d-vision-guided-robotic-bin-picking.jsp>
- [108] Association for Advancing Automation, ‘Smart cameras vs. PC-based machine vision systems’. A3 Vision & Imaging, 2002. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.automate.org/vision/industry-insights/smart-cameras-vs-pc-based-machine-vision-systems/aph>
- [109] Association for Advancing Automation, ‘Smart cameras’. Automation Products. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.automate.org/products/smart-cameras>
- [110] Cognex Corp., ‘Cognex launches highest performance embedded AI vision system powered by qualcomm’. 2026. Accessed: Aug. 18, 2026. [Online]. Available: <https://investor.cognex.com/news/news-details/2026/Cognex-Launches-Highest-Performance-Embedded-AI-Vision-System-Powered-by-Qualcomm/default.aspx>
- [111] Zebra Technologies Corp., ‘Iris GTX smart camera’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.zebra.com/us/en/products/industrial-machine-vision-fixed-scanners/smart-sensors-and-cameras/iris-gtx.html>

- [112] OMRON Corp., ‘FHV7 series smart camera: specifications’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.ia.omron.com/products/family/3740/specification.html>
- [113] SICK AG, ‘InspectorP62x product overview’. 2026. Accessed: Aug. 18, 2026. [Online]. Available: https://www.sick.com/media/productoverview/6/66/066/productoverview_InspectorP62x_g507066_en.pdf
- [114] Teledyne DALSA, ‘BOA pro for industrial inspection applications’. Oct. 20, 2011. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.teledynedalsa.com/en/news/newsroom/press-release-teledyne-dalsa-launches-boa-pro-for-industrial-inspection-applications/>
- [115] LMI Technologies, ‘Gocator 3D smart sensors’. Accessed: Aug. 18, 2026. [Online]. Available: <https://lmi3d.com/3d-smart-sensors/>
- [116] Micro-Epsilon, ‘scanCONTROL configuration tools - easy setup for scanCONTROL laser scanners’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.micro-epsilon.com/2d-3d-measurement/laser-profile-scanners/software/configuration-tools/>
- [117] Baumer Group, ‘AX smart cameras OS/software package’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.baumer.com/int/en/product-overview/industrial-cameras-image-processing/software/baumer-ax-smart-cameras-os-software-package/c/45079>
- [118] Vision Components GmbH, ‘VC-z HALCON image acquisition interface: How to run the vision application inside the camera’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.vision-components.com/fileadmin/external/documentation/software/lib/hacqvcz/a00106.html>
- [119] KEYENCE Corp., ‘Intuitive vision system - CV-x series’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.keyence.com/products/vision/vision-sys/cv-x100/>

- [120] OMRON Corp., ‘FH series vision system: specifications’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.ia.omron.com/products/family/3210/specification.html>
- [121] Cognex Corp., ‘In-sight 6900 series vision controller’. 2026. Accessed: Aug. 18, 2026. [Online]. Available: https://docs.cognex.com/isvs_2610/web/EN/is6900-manual/Default.htm
- [122] Zebra Technologies Corp., ‘4Sight vision controllers’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.zebra.com/us/en/products/industrial-machine-vision-fixed-scanners/vision-controllers/4sight-series.html>
- [123] Zebra Technologies Corp., ‘Aurora design assistant’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.zebra.com/us/en/software/fact-sheets/machine-vision-and-fixed-industrial-scanning-software/aurora-design-assistant.html>
- [124] MVTec Software GmbH, ‘Embedded vision solutions’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.mvtec.com/products/embedded-vision>
- [125] NI, ‘Vision development module’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.ni.com/en/support/downloads/software-products/download.vision-development-module.html>
- [126] Teledyne DALSA, ‘Sherlock supported hardware’. Accessed: Aug. 18, 2026. [Online]. Available: <https://www.teledynedalsa.com/en/products/imaging/vision-software/sherlock/supported-hardware/>
- [127] NVIDIA Corp., ‘Triton inference server - model configuration and model repository’. Accessed: Aug. 18, 2026. [Online]. Available: https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/user_guide/model_configuration.html
- [128] European Parliament and Council of the European Union, ‘Regulation (EU) 2024/1689, art. 72, post-market monitoring by providers and post-market monitoring plan for high-risk AI systems’. 2024. Accessed: Aug. 18, 2026. [Online]. Available: <https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng>

- [129] Amazon Web Services, ‘Real-time inference - amazon SageMaker AI’. Accessed: Aug. 18, 2026. [Online]. Available: <https://docs.aws.amazon.com/sagemaker/latest/dg/realtime-endpoints.html>
- [130] Microsoft, ‘Online endpoints for real-time inference - azure machine learning’. Accessed: Aug. 18, 2026. [Online]. Available: <https://learn.microsoft.com/en-us/azure/machine-learning/concept-endpoints-online?view=azureml-api-2>
- [131] Google Cloud, ‘Vertex AI: endpoints.directRawPredict’. Accessed: Aug. 18, 2026. [Online]. Available: <https://docs.cloud.google.com/vertex-ai/docs/reference/rest/v1/projects.locations.endpoints/directRawPredict>
- [132] LandingAI, ‘LandingLens deployment options’. Accessed: Aug. 18, 2026. [Online]. Available: <https://landinglens.docs.landing.ai/deployment-options>
- [133] LandingAI, ‘Andrew ng - executive chairman and founder’. Accessed: Aug. 18, 2026. [Online]. Available: <https://landing.ai/leadership/andrew-ng>
- [134] Amazon Web Services, ‘Deploy models with amazon SageMaker serverless inference’. Accessed: Aug. 18, 2026. [Online]. Available: <https://docs.aws.amazon.com/sagemaker/latest/dg/serverless-endpoints.html>
- [135] Amazon Web Services, ‘Perform machine learning inference - AWS IoT greengrass’. Accessed: Aug. 18, 2026. [Online]. Available: <https://docs.aws.amazon.com/greengrass/v2/developerguide/perform-machine-learning-inference.html>
- [136] Amazon Web Services, ‘Artificial intelligence and machine learning (AI/ML) - MES on AWS prescriptive guidance’. Accessed: Aug. 18, 2026. [Online]. Available: <https://docs.aws.amazon.com/prescriptive-guidance/latest/mes-on-aws/ai-ml.html>

- [137] Cognex Corp., ‘Cognex OneVision adoption ramps as manufacturers scale AI vision globally’. 2026. Accessed: Aug. 18, 2026. [Online]. Available: <https://investor.cognex.com/news/news-details/2026/Cognex-OneVision-Adoption-Ramps-as-Manufacturers-Scale-AI-Vision-Globally/default.aspx>
- [138] Voxel51, ‘FiftyOne model zoo overview’. Accessed: Aug. 18, 2026. [Online]. Available: https://docs.voxel51.com/model_zoo/overview.html
- [139] Association for Advancing Automation (A3), ‘Components of machine vision systems’. Vision & Imaging Blog, Jun. 28, 2016. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.automate.org/vision/blogs/components-of-machine-vision-systems>
- [140] Cognex Corporation, ‘Vision-guided robotics for industrial automation’. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.cognex.com/en/applications/guidance-and-alignment/vision-guided-robotics-for-industrial-automation>
- [141] IDS Imaging Development Systems, ‘Trigger, flash & co in industrial cameras’. A3 Vision & Imaging Videos, Apr. 23, 2026. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.automate.org/vision/videos/trigger-flash-and-co-in-industrial-cameras>
- [142] Opto Engineering, ‘360° view lenses’. Accessed: Aug. 20, 2026. [Online]. Available: <https://www.opto-e.com/en/products/360-view-lenses>
- [143] Cognex Corporation, ‘Machine vision lighting techniques that actually improve camera performance’. Cognex Resource Center, 2026. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.cognex.com/en/tools-and-resources/resource-center/machine-vision-lighting-techniques>
- [144] S. Wang, R. Veldhuis, C. Brune, and N. Strisciuglio, ‘A survey on the robustness of computer vision models against common corruptions’, *arXiv preprint arXiv:2305.06024*, 2024, doi: [10.48550/arXiv.2305.06024](https://doi.org/10.48550/arXiv.2305.06024).

- [145] A. Silwal, T. Parhar, F. Yandun, and G. Kantor, ‘A robust illumination-invariant camera system for agricultural applications’, *arXiv preprint arXiv:2101.02190*, 2021, doi: [10.48550/arXiv.2101.02190](https://doi.org/10.48550/arXiv.2101.02190).
- [146] R. Szeliski, *Computer vision: Algorithms and applications*, 2nd edn. Cham, Switzerland: Springer, 2022. doi: [10.1007/978-3-030-34372-9](https://doi.org/10.1007/978-3-030-34372-9).
- [147] Association for Advancing Automation (A3), ‘Certified vision professional program’. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.automate.org/vision/vision-certifications/certified-vision-professional>
- [148] Association for Advancing Automation (A3), ‘Course list for the certified vision professional (CVP) basic program’. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.automate.org/vision/certified-vision-professional/cvp-basic-course-list>
- [149] Association for Advancing Automation (A3), ‘Course list for the certified vision professional (CVP) advanced program’. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.automate.org/vision/certified-vision-professional/cvp-advanced-course-list>
- [150] Association for Advancing Automation (A3), ‘Certified system integrator program’. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.automate.org/vision/vision-certifications/certified-system-integrator-program>
- [151] European Machine Vision Association (EMVA), ‘EMVA 1288 certification’. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.emva.org/standards-technology/emva-1288/emva-1288-certification-2/>
- [152] European Machine Vision Association (EMVA), ‘GenICam’. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.emva.org/standards-technology/genicam/>

- [153] International Organization for Standardization, ‘ISO/DIS 24942: Cameras and image sensors used for machine vision applications — Method to measure, compute and present specification parameters and characterization data’. 2026. Accessed: Aug. 19, 2026. [Online]. Available: <https://www.iso.org/standard/88592.html>