International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 15, Issue 4 (Jul. – Aug. 2026), PP. 19-31

Camera-Guided Two-Wheeled Self-Balancing Robots: A State-of-the-Art Review of Nonlinear Modeling, Advanced Control, AI-based Perception, Sensor Fusion and Target Tracking

Tran Thi Kim Thanh
Practice and Experiment Center, Electric Power University, Hanoi, Vietnam

Abstract: *Camera-guided two-wheeled self-balancing robots combine an open-loop unstable, underactuated platform with a perception pipeline that is delayed, intermittent, and sensitive to scene changes. Reliable target following therefore depends on coordinated modeling, estimation, control, visual servoing, tracking, and embedded implementation. This review develops a control-oriented synthesis of recent and foundational work on robot dynamics, PID and LQR stabilization, robust and fuzzy control, MPC and NMPC, learning-assisted control, object detection, multi-object tracking, sensor fusion, and edge deployment. The analysis focuses on interactions between layers rather than isolated algorithm scores. Camera field of view, actuator saturation, timing jitter, target occlusion, confidence calibration, and safe recovery are treated as coupled design constraints. The evidence indicates that transparent deterministic controllers remain strong inner-loop solutions, while MPC-based layers provide a systematic means of enforcing state, input, and visibility limits. Learning methods are most defensible when they support perception, target-motion prediction, bounded adaptation, or task-level decisions around an analyzable stabilizer. A multi-rate architecture is proposed in which time-stamped visual estimates generate bounded motion references that pass through a constraint layer before reaching the balance controller. A staged validation protocol for low-cost platforms is also defined, with reproducibility, variable-delay stability, target-loss recovery, and full-stack latency identified as the main open problems.*

Keywords: *two-wheeled self-balancing robot, inverted pendulum, visual servoing, model predictive control, deep reinforcement learning, object detection, multi-object tracking, sensor fusion, embedded AI, target following*

Date of Submission: 09-07-2026 Date of acceptance: 20-07-2026

I. INTRODUCTION

A two-wheeled self-balancing robot is often introduced as a mobile inverted pendulum. That description is correct, but incomplete for a camera-guided system. The body pitch is open-loop unstable; the forward and pitch dynamics share the same wheel torques, and yaw authority is generated only through differential actuation. Once a camera and target-following task are added, the robot must stabilize itself, estimate its state, interpret a time-varying scene, maintain a target inside the image, and convert image-space errors into dynamically feasible velocity commands. These operations occur at different sampling rates and with different failure modes.

The control literature has progressed from compact linear models and proportional-derivative or linear-quadratic regulators to nonlinear, fuzzy, optimization-based, and data-driven methods [1]-[14]. In parallel, constrained model predictive control and safe learning have made it possible to reason explicitly about saturation, state limits, and uncertain dynamics [15]-[26]. Visual servoing provides the formal bridge between image features and motion [27]-[35], while modern detectors and trackers have changed what can be perceived in real time [36]-[53]. Visual-inertial estimation, simultaneous localization and mapping, camera-LiDAR fusion, model compression, and integer inference now define the practical limits of embedded autonomy [54]-[67]. Active tracking studies on mobile and aerial platforms further expose the importance of viewpoint control and target-loss recovery [68]-[70].

The application space is broad. Civilian uses include indoor security patrol, warehouse escort, inventory observation, telepresence, education, and hazardous-facility inspection. Defence-oriented uses should be framed as non-weapon functions-reconnaissance, perimeter observation, explosive-ordnance-disposal support, logistics, and inspection in environments that are unsafe for personnel. In both domains, the decisive requirement is dependable closed-loop behaviour rather than stand-alone detection accuracy.

The resulting synthesis provides a control-oriented taxonomy that links mechanical modeling, perception, estimation, and embedded implementation. It compares the assumptions and failure modes of major controller families, identifies cross-layer gaps that are not visible in control-only or vision-only surveys, and defines a multi-rate architecture with a validation protocol suitable for a low-cost experimental platform.

II. REVIEW FRAMEWORK AND CONTROL-ORIENTED MODELING

A. Evidence selection and analytical lens

The evidence base was selected using a control-oriented state-of-the-art method rather than a formal meta-analysis. Recent peer-reviewed studies and traceable conference papers were prioritized. Foundational work was retained only when it introduced a model, stability result, visual-servoing taxonomy, learning framework, or estimation architecture that remains essential. Each source was examined against the same closed-loop questions: which plant and actuator model is assumed, which constraints are explicit, how target uncertainty and timing are represented, what failure conditions are tested, and whether the implementation is compatible with the claimed processor and application.

As indicated in Table 1, every contribution was examined through the same closed-loop questions: What plant and actuator model is assumed? Does the controller regulate pitch only, or also translation and yaw? Which constraints are explicit? How is target uncertainty represented? Are end-to-end latency and jitter reported? Do experiments include occlusion, illumination change, floor irregularity, model mismatch, or stale data? Can the method run on a processor compatible with the stated application? This lens prevents misleading comparisons between high-fidelity algorithms tested on workstations and lightweight loops deployed on microcontrollers.

Table 1. Control-oriented taxonomy of the evidence base

Layer	Refs.	Primary question	Recurring limitation
Mechanics and platform	[1]-[14]	How are underactuation, actuation, geometry, payload and terrain represented?	Idealized friction, battery, timing, and camera mass
Advanced control and safe learning	[15]-[26]	How are constraints, uncertainty, learning, and fallback handled?	Compute demand and weak tail-latency evidence
Visual servoing	[27]-[35]	How are image errors mapped into feasible motion?	Limited FOV, depth dependence, and convergence region
Detection and tracking	[36]-[53]	How is a moving target detected and associated over time?	Dataset metrics detached from motion consequences
Fusion and embedded autonomy	[54]-[67]	How are asynchronous sensors and limited compute managed?	Calibration drift, timing jitter, and thermal effects
Active target tracking	[68]-[70]	How are perception and viewpoint control closed in one loop?	Safety, transfer, and recovery evidence

B. Nonlinear dynamics, actuation, and timing

The minimum useful model contains wheel rotation, body pitch, planar translation, and yaw. Let the generalized coordinate vector be $q = [x, y, \psi, \theta, \varphi_L, \varphi_R]^T$, where θ is body pitch and ψ is yaw. The nonlinear rigid-body dynamics can be written in the standard form

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{f}(\dot{\mathbf{q}}) = B\boldsymbol{\tau} + \mathbf{d} \tag{1}$$

where M is the configuration-dependent mass matrix, C contains Coriolis and centrifugal terms, g is gravity, f represents friction and other dissipative effects, B maps left and right wheel torques into generalized forces, and d collects disturbances and neglected dynamics. Tire compliance, backlash, payload motion, cable forces, and camera-mast vibration are particularly relevant on low-cost prototypes [1]-[7].

For wheel radius r and half track b, the nominal longitudinal force and yaw moment are

$$F_x = \frac{\tau_R + \tau_L}{r} \tag{2a}$$

$$M_z = \frac{b(\tau_R - \tau_L)}{r} \tag{2b}$$

The sum-difference decomposition does not decouple the plant. Acceleration requires the body to lean, turning changes contact forces, and a camera above the axle raises the center of mass. Camera placement is therefore both an observability and a stability decision.

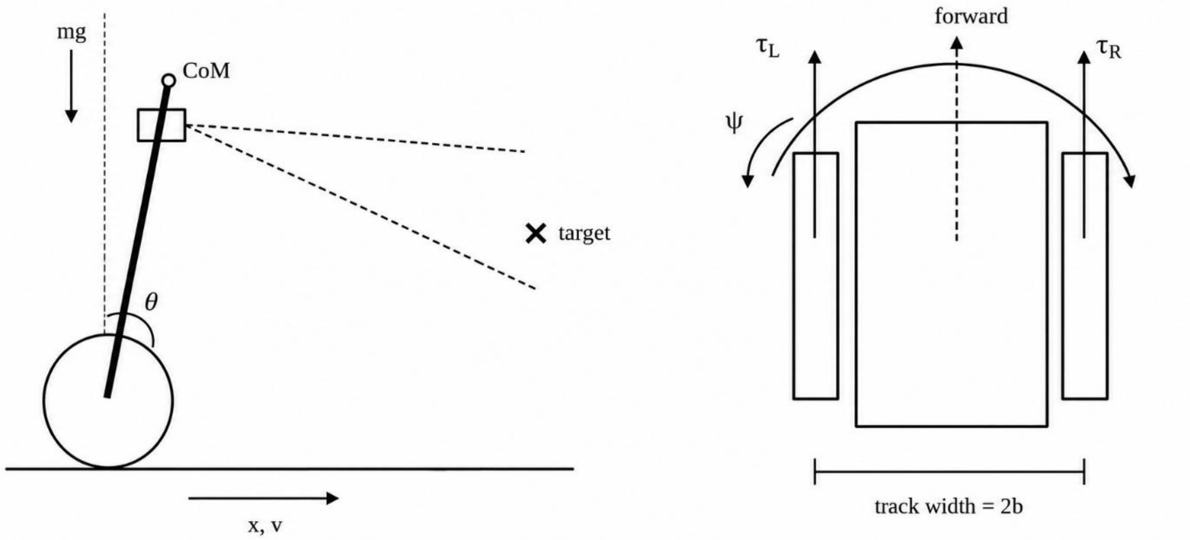

Fig. 1. Control-oriented geometry of a camera-guided two-wheeled self-balancing robot

Around the upright equilibrium, a compact scheduled model can be used for controller synthesis:

$$\begin{cases} \dot{\mathbf{x}}_s = A(\rho)\mathbf{x}_s + B(\rho)\mathbf{u} + E(\rho)\mathbf{w} \\ \mathbf{y} = C\mathbf{x}_s + \mathbf{v} \end{cases} \tag{3}$$

Here $x_s = [x, \dot{x}, \theta, \theta, \psi, \dot{\psi}]^{\mathrm{T}}$ and ρ may represent battery voltage, payload, or another operating variable. Linearization is scientifically acceptable when its validity region is identified and the model includes actuator lag, dead zone, saturation, rate limits, wheel-speed limits, quantization, and sensor-to-actuator delay. In a camera-guided system, the delay is time varying because frame acquisition, inference, association, messaging, and task scheduling introduce jitter. Tail latency, not only mean latency, must therefore enter simulation and stability tests.

Parameter identification should separate quantities that can be measured directly from those inferred dynamically. Wheel radius, track width, mass, center-of-mass height, motor resistance, and gear ratio can be measured or obtained from specifications. Inertia, viscous friction, Coulomb friction, dead zone, and effective torque constants should be estimated from dedicated experiments. Step, chirp, and free-decay tests should be repeated at several battery voltages and payloads so that uncertainty intervals, rather than one nominal parameter set, enter validation.

A two-model practice is recommended. The online controller uses a compact model that can be analyzed or optimized within its deadline. A separate validation model includes nonlinear geometry, motor and battery effects, friction, quantization, rolling resistance, frame loss, camera timing, and parameter distributions. Agreement between these models is not assumed. The role of the higher-fidelity model is to expose fragility in the controller design model.

C. Camera geometry and target-state interface

A monocular follower can use target-box center u_c and box height h_c as compact measurements. With image principal point u_0, horizontal focal length f_x, and desired box height h_d, calibration-aware visual errors are

$$e_u = \frac{u_c - u_0}{f_x}, \qquad e_h = \ln\left(\frac{h_d}{h_c}\right) \tag{4}$$

The horizontal error approximates target bearing for moderate angles, while the logarithmic scale error is a symmetric surrogate for approach and retreat. A bounded outer loop can then generate forward-speed and yaw-rate references:

$$v_{\mathrm{ref}} = \mathrm{sat}_{[-v_{\max}, v_{\max}]}(k_h e_h), \qquad \omega_{\mathrm{ref}} = \mathrm{sat}_{[-\omega_{\max}, \omega_{\max}]}(-k_u e_u) \tag{5}$$

The negative yaw sign assumes positive image displacement to the right and positive robot yaw to the left. This convention must be stated consistently in implementation. Saturation and acceleration limiting prevent a box jump or false association from driving the balance loop outside its admissible envelope. When confidence is low, the measurement is stale, or the target leaves the FOV, the robot must enter an explicit coast, search, or stop mode. Table 2 summarizes the minimum modeling and interface requirements for reliable balance control and camera-based target following.

Table 2. Model and interface requirements for closed-loop target following

Element	Minimum representation	Control consequence	Frequent omission
Pitch-translation coupling	Nonlinear model plus upright linearization	Defines balance authority during acceleration	Independent position-loop assumption
Yaw and contact	Differential torque, slip/turning limits	Bounds image-centering motion	Perfect rolling at all speeds
Motor and battery	Lag, dead zone, voltage and rate limits	Sets bandwidth and recovery from saturation	Constant torque gain
Camera and mount	Intrinsics, extrinsics, FOV, vibration, added inertia	Links visibility and body motion	Massless, delay-free camera
Timing	Exposure time, inference delay, jitter, frame loss	Sets phase margin and estimator consistency	Average FPS only
Target state	Bearing, scale/depth, velocity, covariance, confidence, age	Supports prediction and safe mode logic	Stationary or perfectly observed target

III. ADVANCED BALANCE AND MOTION CONTROL

A. Classical, optimal, fuzzy, and robust stabilization

Cascaded PID remains attractive because it is transparent, inexpensive, and readily executed at several hundred hertz. Strong implementations separate a fast pitch loop from slower velocity and position loops, add anti-windup, filter derivative action, and limit reference acceleration. On a well-identified robot with a narrow payload range, disciplined PID can outperform a more complex method whose model or timing assumptions are violated [4]-[7].

LQR offers a systematic trade-off among pitch, velocity, position, yaw, and control effort. It exposes controllability and closed-loop poles, and it can be extended with integral action or gain scheduling. Its limitation is local validity. Large disturbances, aggressive visual reorientation, payload variation, and contact changes can move the plant outside the identified operating region. Fuzzy logic or neural tuning can soften nonlinear friction and schedule gains, but their rule bases, scaling factors, and output bounds must be disclosed. The safest use is bounded gain scheduling or residual compensation around a nominal controller rather than unrestricted torque generation.

Sliding-mode, backstepping, feedback-linearization, and H-infinity methods address uncertainty more explicitly. Their benefit depends on realistic disturbance bounds and actuator bandwidth. Chattering, sensor-noise amplification, and neglected motor dynamics can defeat nominal robustness. Boundary layers, higher order sliding modes, and disturbance observers reduce these effects, but they add design parameters and delay. Wheel-legged platforms extend the terrain envelope through whole-body and virtual-model control [8]-[14], although their contact and kinematic complexity exceeds that of a rigid TWSBR.

B. MPC, NMPC, and learned models

MPC is well matched to camera-guided balancing because pitch, wheel speed, voltage, acceleration, and FOV limits can be represented in the same finite-horizon problem. A representative objective is

$$J = \sum_{i=0}^{N-1} \left[\| \mathbf{x}_{k+i|k} - \mathbf{x}^\star \|_Q^2 + \| \mathbf{e}_{k+i|k}^{v} \|_{Q_v}^2 + \| \mathbf{u}_{k+i|k} \|_R^2 + \| \Delta\mathbf{u}_{k+i|k} \|_S^2 \right] + \| \mathbf{x}_{k+N|k} - \mathbf{x}^\star \|_P^2 \tag{6}$$

where the state-tracking term preserves balance and motion, e^{v} denotes visual error, Δu penalizes command variation, and P defines the terminal penalty. The optimization is subject to

$$\mathbf{x}_{k+i+1|k} = f(\mathbf{x}_{k+i|k}, \mathbf{u}_{k+i|k}), \quad \mathbf{x}_{k+i|k} \in \mathcal{X}, \quad \mathbf{u}_{k+i|k} \in \mathcal{U}, \quad \Delta\mathbf{u}_{k+i|k} \in \Delta\mathcal{U}, \quad \mathbf{e}_{k+i|k}^{v} \in \mathcal{E}_{\mathrm{FOV}} \tag{7}$$

Linear MPC is efficient near the upright state. NMPC captures larger-angle dynamics and camera geometry, but its value is conditional on meeting every solver deadline on the intended processor. Median solve time is insufficient. Reports should include the 95th and 99th percentiles, the maximum, the infeasibility rate, the warm-start policy, and the backup action. Learning-based MPC can estimate residual dynamics or changing parameters [15], [16], [24], [25], but uncertainty must enter the constraint tightening. A predictor that improves average one-step accuracy while extrapolating unpredictably outside the training distribution is not a safe model.

C. Deep reinforcement learning and safety-constrained hybrids

DRL learns long-horizon policies from high-dimensional observations, yet direct image-to-torque control remains difficult to certify and sample inefficient. DQN, DDPG, PPO, and SAC provide standard value-based and continuous-control foundations [20]-[23], while model-based and information-theoretic methods improve sample efficiency [24], [25]. For a self-balancing robot, DRL is most credible at the slower task level: selecting a target-following mode, predicting target motion, scheduling references, or generating a bounded residual around an analyzable stabilizer.

A predictive safety filter formalizes this separation. Given an AI proposal u_k^{AI}, the filter computes the closest command whose predicted trajectory remains in a declared safe set:

$$\mathbf{u}_k^{\text{safe}} = \arg\min_{\mathbf{u}} \ \| \mathbf{u} - \mathbf{u}_k^{\text{AI}} \|_W^2 \quad \text{s.t.} \quad \mathbf{x}_{k+j|k} \in \mathcal{X}_{\text{safe}},\ \mathbf{u}_{k+j|k} \in \mathcal{U},\ j = 0, \dots, N_s \tag{8}$$

If the problem is infeasible or the data are stale, a documented backup action is invoked. This arrangement aligns AI use with the underlying research objective. Perception and adaptation improve capability, while deterministic balance and constraint enforcement preserve interpretable safety [16]-[18]. Domain randomization should include mass, center of gravity, motor gain, friction, battery voltage, camera delay, blur, target motion, packet loss, and processor jitter. Randomizing texture alone does not test the coupled dynamics.

Table 3. Control methods for camera-guided TWSBR

Method	Strength	Primary limitation	Recommended role	Evidence required
Cascaded PID	Low compute, transparent, reliable fallback	Gain sensitivity, weak multivariable constraints	Fast inner loop and baseline	Margins, anti-windup, delay sweep
LQR/LQI	Systematic state trade-off and pole analysis	Local model and estimator dependence	Inner balance/velocity/yaw regulator	Operating envelope and gain schedule
Fuzzy/neural tuning	Adapts gains or residuals to nonlinear effects	Rule/weight opacity, weak guarantees	Bounded adaptation	Ablation and output bounds
Robust nonlinear	Strong rejection under stated uncertainty	Noise, chattering, actuator demand	High-disturbance stabilization	Torque rate and bandwidth tests
MPC	Explicit multivariable constraints	Model and solver dependence	Reference governor or motion layer	Tail solving time and infeasibility
NMPC	Nonlinear dynamics and FOV constraints	High computational burden	Integrated vision-motion control	On-target timing and fallback
DRL	Long-horizon task policy and complex observations	Sample cost, transfer, weak guarantees	Mode selection and prediction	Seeds, transfer protocol, failure set
AI + safety filter	Adaptation with declared limits	Interface and architecture complexity	Preferred hybrid structure	Safe set, backup and recovery tests

Controller comparisons should use identical actuator limits, reference filters, estimator outputs, and disturbance realizations (see Table 3). Otherwise, a method can appear superior because it is allowed more torque, receives cleaner states, or is evaluated at a slower command. At minimum, the comparison should report pitch RMS and peak, settling time, wheel-speed and voltage peaks, input variation, energy, saturation duration, and the largest recoverable push or initial angle.

Stability claims must match the implemented architecture. A continuous-time Lyapunov proof for a nominal inner loop does not establish stability of a sampled, saturated, delay-varying camera-guided system. The strongest evidence combines local analytical guarantees, delay and gain-margin analysis, constraint verification, and hardware fault injection. Learning components require a separate statement of the domain in which their output remains bounded and the condition under which the nominal controller resumes authority.

IV. VISUAL PERCEPTION, TARGET TRACKING, AND SENSOR FUSION

A. Visual servoing and learned visual control

Classical visual servoing defines a task error from image features or reconstructed pose. IBVS uses image-space errors and is generally robust to calibration errors, but its interaction matrix depends on depth and large motions may push features outside the FOV. PBVS controls an estimated three-dimensional pose and generates intuitive geometric trajectories but pose and calibration errors enter the feedback loop. Hybrid schemes combine the two formulations [27]-[31].

Learned visual servoing replaces hand-designed features with representations, transformations, or policies. Bateux et al. inferred control-relevant transformations from images [32]. Subsequent methods combine DRL with optimal control to enlarge convergence while retaining a precise local controller [33], use multi-perspective latent actions [34], or predict future images in a visual MPC loop [35]. These approaches are promising, but a compact interface-bearing, scale or depth, image-plane velocity, covariance, confidence, and age-is often more robust and easier to validate on a low-cost TWSBR than raw-pixel torque control.

B. Detection, association, and target-loss recovery

Two-stage detectors such as Faster R-CNN offer strong localization but are often too heavy for a small edge processor [36]. One-stage models including YOLO, SSD, focal-loss detectors, and EfficientDet made real-time embedded detection practical [37]-[41]. DETR and later deformable or real-time variants model global context and simplify parts of the detection pipeline [42], [43], [46]. YOLOX and YOLOv7 emphasize efficient training and real-time accuracy [44], [45], while the broader YOLO evolution is surveyed in [47].

Closed-loop selection must consider box-center stability, scale noise, small-target recall, confidence calibration, and latency distribution-not only mean average precision. Center noise appears as yaw-command noise, scale noise excites forward acceleration, and intermittent misses trigger mode changes. Tracking-by-detection improves temporal consistency. Deep SORT, Tracktor, FairMOT, CenterTrack, ByteTrack, and BoT-SORT offer different combinations of motion, appearance, and detector reuse [48]-[53]. For a single followed target, safe association and recovery logic matter more than leaderboard MOTA alone.

A practical supervisor includes TRACK, COAST, SEARCH, STOP, and REACQUIRE. Transition thresholds should depend on confidence, covariance, time since the last valid observation, obstacle distance, and available balance margin. Motion prediction can bridge a short occlusion, but covariance must grow with prediction horizon. When uncertainty becomes large, deceleration or stop is preferable to confident open-loop pursuit.

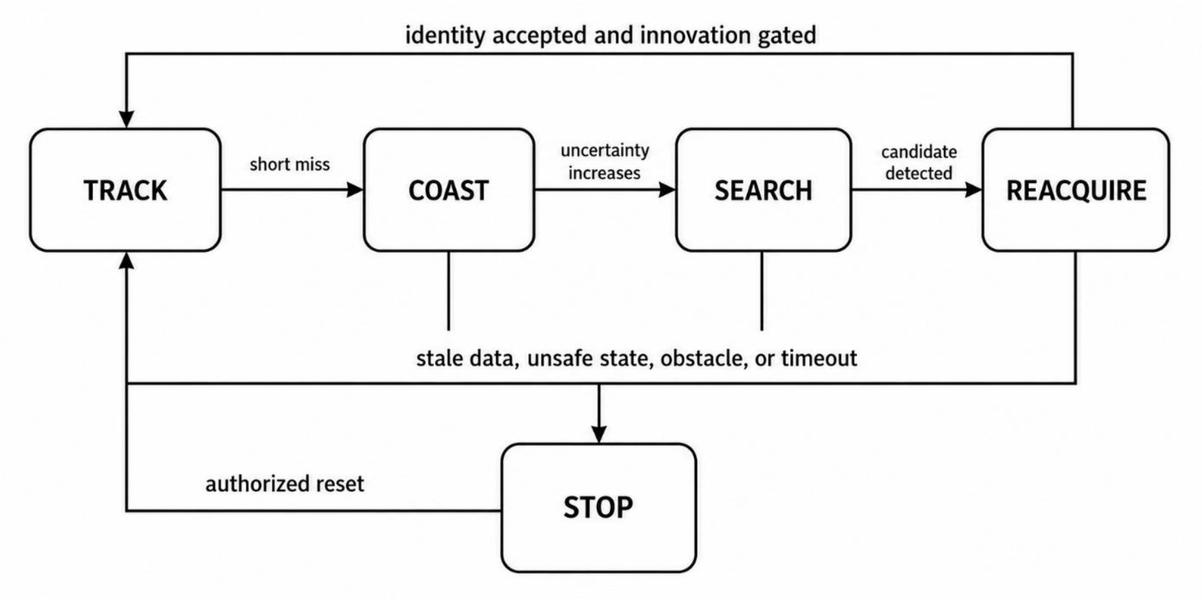

Fig. 2. Confidence- and covariance-aware target-loss supervisor

Detector and tracker evaluation should therefore include control-sensitive quantities: box-center spectral content, scale bias, confidence reliability, false reacquisition rate, target-loss duration, and time to a safe stop. These quantities should be measured at the deployed camera height and during robot motion. Public-dataset accuracy is valuable for model selection, but it cannot replace moving-platform sequences with blur, low-angle appearance, occlusion, reflective floors, and distractor crossings. Figure 2 illustrates a confidence- and covariance-aware supervisory logic that manages target loss through staged coast, search, reacquisition, and fail-safe stopping modes.

C. Time-aligned fusion and embedded implementation

The inner controller normally uses a fused pitch estimate from gyroscope and accelerometer data, while wheel encoders provide velocity and displacement. A complementary filter is adequate for many platforms. EKF or UKF formulations estimate bias and propagate covariance. For a nonlinear time-stamped measurement model, a generic EKF cycle is

$$\begin{gathered}\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}) \\ P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \\ K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \\ \hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k[\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})] \\ P_{k|k} = (I - K_k H_k) P_{k|k-1}\end{gathered} \tag{9}$$

Camera data belong to exposure time, not inference-completion time. A consistent estimator updates a buffered past state and re-propagates it or augments the state with delay. MSCKF, OKVIS, VINS-Mono, ORB-

SLAM2/3, and LIO-SAM demonstrate the importance of exact time stamps, online or repeated calibration, outlier gating, and covariance-aware fusion [55]-[60]. Full SLAM is not mandatory for every indoor follower, but these architectural principles remain valid (see Table 4).

Clock synchronization and calibration should be treated as experimental variables. Camera-to-IMU time offset, rolling-shutter readout, encoder timestamp granularity, and extrinsic drift all appear as structured estimation error. Innovation sequences and normalized innovation squared tests can reveal inconsistency. Controller authority should decrease when the estimator reports large covariance or repeated gated measurements.

Object-level fusion can add depth or obstacle information. TransFuser, BEVFusion, and DeployFusion illustrate camera-range fusion and deployable representations [61]-[63]. The trade-off is additional power, compute, calibration, and top-mounted mass. Edge deployment is likewise a co-design task: the detector, tracker, estimator, controller, communication, and logging compete for memory bandwidth and processor time. Embedded tracking [64], MobileNetV2 [65], integer-only quantization [66], and compression [67] reduce workload, but quantized models must be checked for confidence and box stability, not only aggregate accuracy.

The balance loop should run in a fixed-priority real-time task, read the latest validated reference atomically, and continue safely when perception misses deadlines. Vision can run asynchronously and publish time-stamped target states. A watchdog invalidates stale data and initiates controlled stopping. Timing reports should separate exposure/readout, preprocessing, inference, association, messaging, estimation, control, and actuator update, and should report median, 95th percentile, 99th percentile, maximum, and drop rate under thermal steady state.

Table 4. Perception, fusion and embedded options from a control perspective

Class	Representative refs.	Benefit	Control risk	Embedded guidance
Two-stage detector	[36]	Accurate localization and difficult examples	High latency and memory	Use only with adequate acceleration
One-stage CNN detector	[37]-[41], [44], [45]	Low latency and mature tooling	Box jitter and small-target trade-off	Profile the complete batch-one pipeline
Transformer detector	[42], [43], [46]	Global context and end-to-end set prediction	Memory and training demand	Prefer compact real-time variants
Appearance-assisted tracker	[48], [50], [53]	Identity retention through crossings	Embedding cost and domain sensitivity	Use when distractors are common
Motion-centric tracker	[49], [51], [52]	Fast association and short-occlusion recovery	Drift after long disappearance	Pair with covariance-aware supervisor
VIO/SLAM	[55]-[60]	Time-aligned motion and uncertainty	Calibration and computational burden	Use only to the mission-required fidelity
Camera-range fusion	[61]-[63]	Depth and obstacle observability	Power, mass, and calibration burden	Select by terrain and safety need
Compression/quantization	[64]-[67]	Lower latency and power	Confidence and localization changes	Revalidate closed-loop target tracking

V. INTEGRATED ARCHITECTURE, APPLICATIONS, AND RESEARCH DIRECTIONS

Figure 3 synthesizes the preferred architecture. The inner balance and yaw controller is deterministic and independent of camera deadlines. A target-state estimator receives time-stamped detector and tracker outputs and produces bearing, scale or depth, velocity, covariance, confidence, and age. The visual task controller generates bounded velocity references. A constraint layer verifies those references against pitch, wheel speed, acceleration, FOV, obstacle, and freshness limits before they reach the stabilizer. AI may improve perception, prediction, mode selection, or bounded residual adaptation, but it cannot bypass the constraint layer.

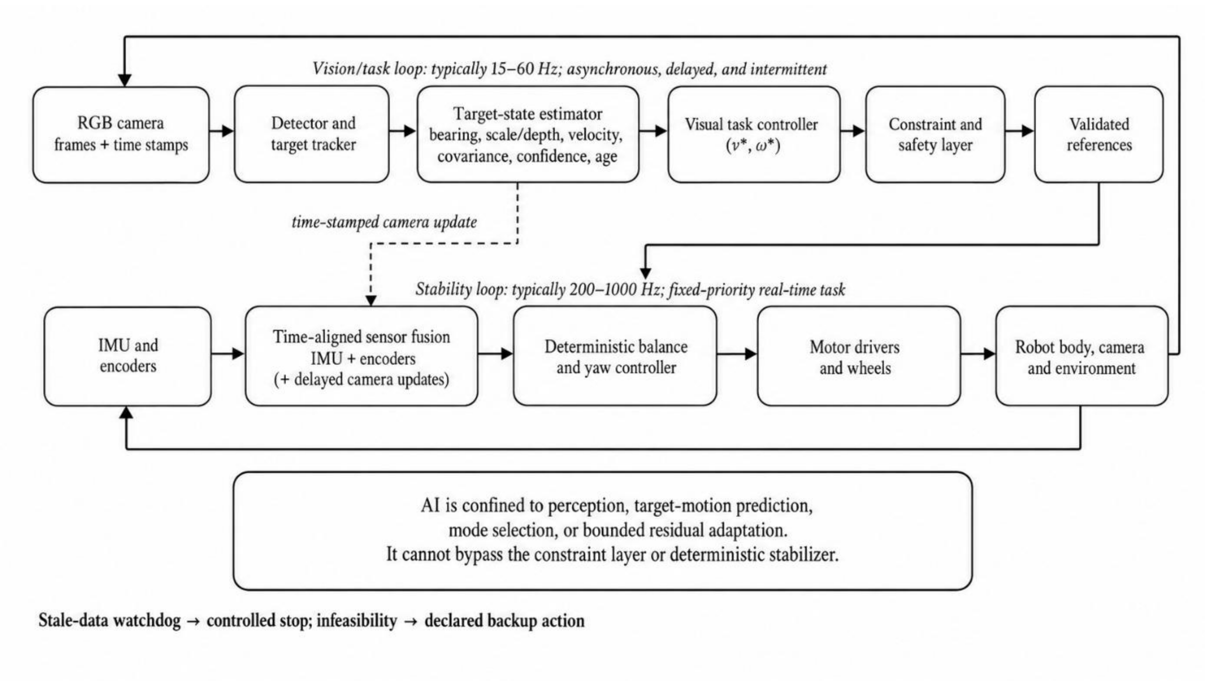

Fig. 3. Multi-rate closed-loop architecture for a camera-guided two-wheeled self-balancing robot

Warehouse and factory following is partly structured, yet shelves cause occlusion, polished floors change traction, and wireless congestion affects teleoperation. Mobile inspection and telepresence add camera-quality requirements: a small pitch RMS can still produce blur if the controller excites a flexible mount. Exposure-aware speed limits, trajectory smoothing, and optional gimbal stabilization improve delivered imagery. Mission energy matters because aggressive balancing and repeated search reduce endurance.

For civil security and non-weapon defence observation, the same platform can support perimeter monitoring, reconnaissance, EOD inspection, hazardous-material assessment, and logistics. Active-tracking studies show how viewpoint control can be optimized or learned [68]-[70]. Operational acceptance additionally requires authenticated software, encrypted communication, audit logs, privacy controls, geofencing, local safe stop, and human-authorized mission rules. Detector confidence alone must never authorize a consequential action.

Terrain should determine the platform choice. A rigid TWSBR is efficient on flat surfaces but vulnerable to steps, soft ground, and abrupt slopes. Wheel-legged designs extend the operating envelope [8]-[14] at the cost of mass, hardware, control complexity, and power. The research platform should therefore begin with the low-cost rigid two-wheel system described in the doctoral direction and treat wheel-legged mechanisms as a later extension rather than mixing the two validation problems.

The literature shows substantial progress in each component but limited system-level integration. Balance-control studies often assume perfect state measurements and ignore camera computation. Vision studies optimize detector or tracker metrics without propagating box uncertainty into motion. Learning studies emphasize nominal success while omitting fallback behaviour. Edge studies report throughput but not stability. The principal research gaps are therefore cross-layer rather than algorithm-specific. Table 5 identifies the main cross-layer research gaps and the actions and evidence required to address them.

Table 5. Cross-layer gaps and evidence needed to close them

Gap	Why evidence remains insufficient	Research action	Minimum evidence
Unified dynamics-vision benchmark	Mechanics, vision, and timing are validated separately	Publish compact/high-fidelity models, identified parameters, timing traces	Reproducible simulation and hardware identification
Variable-delay stability	Visual feedback is asynchronous and intermittent	Use sampled data/hybrid analysis and injected delay/dropout	Bounded pitch under declared tail delays
Safety-aware learning	Nominal reward hides unsafe transients	Combine policy with MPC/CBF/safety filter and backup	No hard-limit violations in predeclared tests

Gap	Why evidence remains insufficient	Research action	Minimum evidence
Occlusion and reacquisition	Tracking metrics omit motion consequences	Use confidence/covariance supervisor and controlled stopping	Loss duration, stop distance, reacquisition success
Edge co-design	FPS hides scheduling and thermal effects	Profile full stack under sustained load	Latency distribution and thermal steady state
Joint reporting	Control and vision use separate protocols	Adopt mission-level control-perception scorecard	Raw logs, seeds, repeated trials, failure cases
Security and oversight	Remote observation creates attack and privacy risks	Local safe stop, authentication, audit and human authority	Network-loss and unauthorized-command tests

The validation sequence in Fig. 4 proceeds from physical identification and a measured baseline to vision, constraint handling, and AI augmentation. This order prevents a learning component from masking an unresolved model or timing error. Each stage requires a controlled ablation that begins with a physics-only stabilizer and then adds perception, constraint handling, prediction, and the complete hybrid architecture.

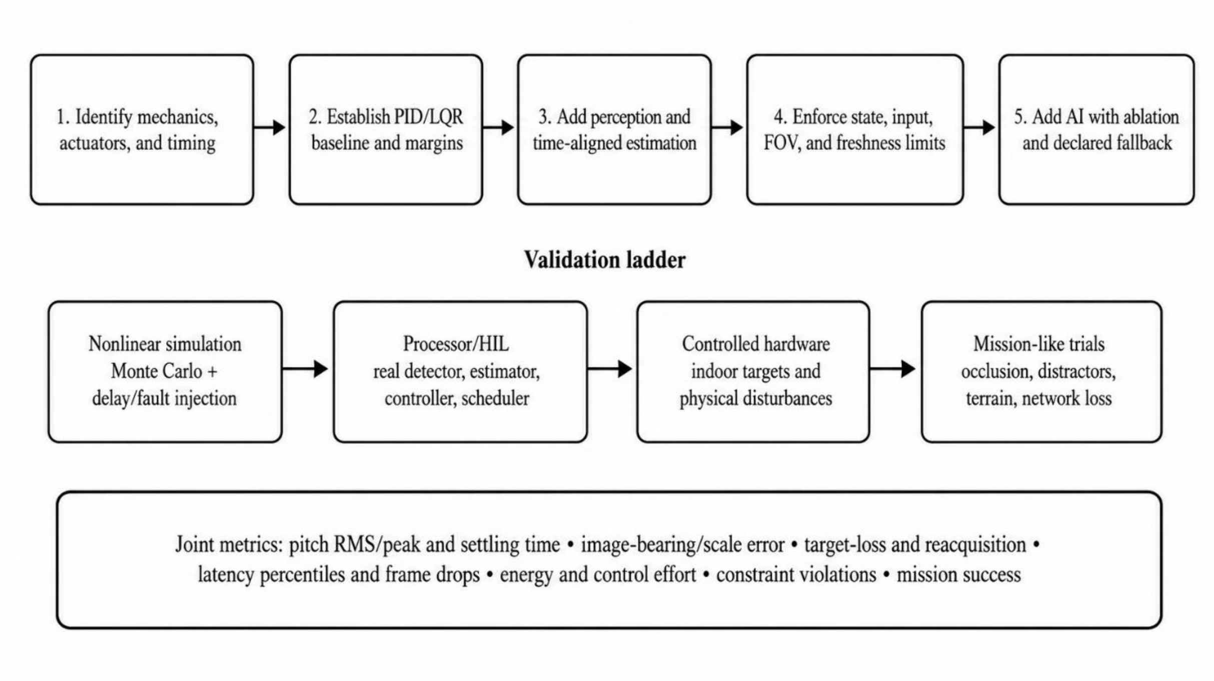

Fig. 4. Evidence-building and validation ladder for integrated control and vision

Simulation should include constant-velocity, stop-and-go, lateral-crossing, and abrupt-acceleration targets, together with push disturbances, payload change, slope or friction variation, delayed frames, and packet loss. Mechanical and visual parameters should be varied jointly in Monte Carlo trials. Processor-in-the-loop or hardware-in-the-loop tests should execute the actual detector, estimator, control software, and scheduler before free-running experiments. Real trials should then expand from controlled indoor scenes to illumination change, patterned backgrounds, partial occlusion, distractors, uneven floors, and network interruption.

A single composite score can hide unsafe trade-offs. Results should instead report Pareto relationships: mission success versus constraint violations, tracking error versus energy, and detector complexity versus tail latency. Repeated-run confidence intervals and multiple random seeds are required for learned methods. Failure cases and termination conditions belong in the main evidence, not only in supplementary video.

A practical low-cost program can be staged on a rigid TWSBR equipped with an IMU, wheel encoders, an RGB camera, and a widely available embedded processor. The first hardware milestone is stable balance with measured margins and timing. The second is stationary-target tracking. Moving targets, occlusion, distractors, and lighting variation follow only after the estimator and safety supervisor are validated. This progression preserves a clear causal link between each added capability and the observed change in performance (see Fig. 4 and Table 6).

Table 6. Minimum reporting checklist for reproducible studies

Item	Minimum disclosure	Why it matters	Evidence
Mechanics/actuation	Mass, inertia, geometry, gear ratio, motor limits, friction, payload	Defines instability and available torque	Identified parameters and uncertainty ranges
Camera/perception	Intrinsics/extrinsics, exposure, resolution, model version, thresholds	Defines FOV, noise, delay, target loss	Calibration and representative sequences
State estimation	State vector, equations, covariances, time stamps, gating	Links asynchronous sensing and control	Innovation, drift, and delay-injection tests
Controller/constraints	Rates, gains/weights, saturation, safe set, fallback	Determines stability, feasibility, recovery	Operating envelope and violation count
Timing/computation	Processor, precision, priorities, tail latency, drops, temperature	Determines phase lag and deadline consistency	Raw timing log and sustained-load test
Scenarios/disturbances	Targets, occlusion, distractors, lighting, push, slope, payload	Tests coupled robustness	Predeclared scenarios and repeated runs
Metrics/statistics	Pitch, energy, image error, loss/reacquisition, mission success	Prevents selective component reporting	Intervals, Pareto plots, run-level data
Safety/security	Watchdog, stop, network loss, authentication, human override	Limits consequence of faults or misuse	Fault-injection and recovery records
Reproducibility	Source, models, weights, calibration, versions, configuration	Enables independent comparison	Persistent repository and execution steps

Table 7 converts the proposed ladder into a minimum experimental matrix. Numerical acceptance thresholds should be declared before testing and scaled to the platform rather than copied across robots with different mass, wheel radius, motors, and camera geometry. The evidence structure must remain invariant across nominal tracking, coupled disturbances, perception failure, timing failure, and safe terminal response.

Table 7. Recommended experimental scenarios and acceptance measures

Scenario	Input or disturbance	Primary measures	Acceptance focus
Upright recovery	Initial pitch offsets and calibrated pushes	Pitch peak/RMS, recovery time, wheel speed	Return without hard-limit violation
Speed and position	Steps, ramps, reversals, reference-rate limits	Tracking error, overshoot, input variation, energy	Stable response within declared envelope
Yaw/visual centering	Stationary target at multiple bearings and distances	Image error, yaw response, box-center jitter	Target retained inside FOV
Moving target	Constant speed, stop-and-go, crossing, acceleration	Bearing/scale error, following distance, mission success	No balance loss during pursuit
Occlusion/distractors	Timed occlusions, crossing identities, re-entry	Loss duration, false association, stop distance, reacquisition	No pursuit of an unverified identity
Timing fault	Injected delay distribution, dropped frames, stale packets	Tail latency, pitch response, command age	Controlled degradation and watchdog action
Surface/payload	Friction change, slope, uneven floor, mass shift	Stability margin, control effort, tracking degradation	Remain inside published operating region
Thermal/network	Sustained inference load and link interruption	Throttling, latency drift, local-stop response	Local safety independent of remote link

For each scenario, raw time histories should be preserved at the native sampling rate so that control, vision, and scheduling events can be aligned. Summary statistics should be accompanied by representative failures, not only successful runs. This requirement is particularly important when confidence thresholds, tracker states, or safety-filter feasibility change during a trial.

A focused doctoral program can implement this agenda through three nested time scales. The fastest layer estimates pitch and regulates wheel torque from the IMU and encoders. The intermediate layer coordinates forward velocity and yaw while enforcing actuator, tilt, and acceleration constraints. The slowest layer processes camera frames, estimates target state, and selects a tracking or recovery mode. Separating these layers permits independent timing tests and prevents camera workload from pre-empting the balance loop. It also allows the perception model to be replaced without changing the safety-critical interface.

The first control contribution should be established against transparent PID and LQR baselines on the same identified plant. A constrained MPC or NMPC layer can then be introduced as a reference governor or motion controller, with pitch, wheel speed, voltage, acceleration, and FOV represented explicitly. AI should enter only after this constrained baseline is stable. Suitable roles include target-motion prediction, online scheduling of model parameters or weights, and task-level mode selection. A reinforcement-learning policy may propose a desired velocity or mode, but the admissible command remains the output of the model-based constraint layer.

The perception contribution should likewise be framed as a closed-loop problem. Detector selection must be based on stable target centers, distance-proxy quality, confidence calibration, and tail latency on the chosen embedded processor. Tracking should produce covariance and data age in addition to an identity. These signals feed the TRACK-COAST-SEARCH-STOP-REACQUIRE supervisor, so their calibration can be assessed by motion consequences: false acceleration, unnecessary stopping, target-loss duration, and false reacquisition. Camera calibration, mount vibration, rolling shutter, and exposure control should be included because they influence both perception quality and controller excitation.

Ablation is essential for attributing improvement. The minimum sequence comprises five configurations: the stabilizer alone, the stabilizer with visual measurements but no prediction, the constrained visual controller, the constrained controller with target prediction, and the complete AI-assisted system. Each configuration should use identical target trajectories, disturbances, processor settings, and safety limits. Learned components should be evaluated across multiple random seeds and held-out physical conditions. The deterministic baseline should be retuned only according to a predeclared procedure. This design distinguishes a genuine architectural gain from extra computation, favorable tuning, or a less demanding test.

The final evidence should connect low-cost feasibility with scientific generality. Experiments on an IMU-encoder-RGB platform demonstrate deployability, but the report should state which conclusions depend on a particular camera, processor, motor, or floor. Parameter sweeps and dimensionless quantities, such as normalized pitch error, torque utilization, and latency relative to the inner-loop period, help transfer findings to other platforms. Publishing time-aligned logs, model parameters, calibration files, controller settings, and failure cases would turn the platform into a reusable benchmark rather than a one-off demonstration.

VI. CONCLUSION

Camera-guided two-wheeled self-balancing robots lie at a demanding intersection of nonlinear control, perception, estimation, and embedded computing. The reviewed literature shows that no single algorithm dominates every layer. PID and LQR remain valuable because they are fast, transparent, and straightforward to validate. Robust nonlinear methods improve disturbance rejection when noise and actuator limits are respected. MPC and NMPC provide the most direct treatment of coupled state, input, timing, and FOV constraints. Deep learning has transformed detection and tracking, while reinforcement learning and learned models offer adaptation and long-horizon task decisions.

The strongest design pattern is hierarchical rather than fully end to end. A deterministic inner loop preserves balance. A time-aligned estimator reconciles IMU, encoder, and camera information. An outer visual controller generates bounded references, and a constraint layer blocks learned or noisy commands that violate the feasible envelope. This architecture is compatible with low-cost hardware and makes faults observable.

Future research should prioritize reproducible dynamics-vision benchmarks, stability under variable delay and dropout, confidence-aware target-loss recovery, safety-constrained learning, and joint control-perception metrics. For civilian and non-weapon defence observation, cybersecurity, privacy, and human oversight are engineering requirements. Progress should be judged by dependable mission completion under realistic timing, visibility, terrain, and uncertainty-not by detector accuracy or nominal pitch error in isolation.

REFERENCES

[1] F. Grasser, A. D'Arrigo, S. Colombi, and A. C. Rufer, "JOE: A mobile, inverted pendulum," IEEE Transactions on Industrial Electronics, vol. 49, no. 1, pp. 107-114, 2002. https://doi.org/10.1109/41.982254

[2] K. Pathak, J. Franch, and S. K. Agrawal, "Velocity and position control of a wheeled inverted pendulum by partial feedback linearization," IEEE Transactions on Robotics, vol. 21, no. 3, pp. 505-513, 2005. https://doi.org/10.1109/TRO.2004.840905

[3] O. Boubaker, "The inverted pendulum: A fundamental benchmark in control theory and robotics," in Proc. IEEE International Conference on Education and e-Learning Innovations, 2012. https://doi.org/10.1109/ICEELI.2012.6360606

[4] M. M. Rahman, S. M. H. Rashi, K. M. R. Hassan, and M. M. Hossain, "Comparison of different control theories on a two-wheeled self-balancing robot," AIP Conference Proceedings, vol. 1980, no. 1, Art. no. 060005, 2018. https://doi.org/10.1063/1.5044373

[5] M. A. A. Morshed, M. M. Hayder, and T. R. Maruf, "Electromechanical system design for self-balancing robot," arXiv:2310.12142, 2023. https://arxiv.org/abs/2310.12142

[6] A. Abdelgawad, T. Shohdy, and A. Nada, "Model- and data-based control of self-balancing robots: Practical educational approach with LabVIEW and Arduino," IFAC-PapersOnLine, Volume 58, Issue 9, 2024. https://doi.org/10.1016/j.ifacol.2024.07.399

[7] K. Laddach, M. Czyżniewski, and R. Łangowski, "A selection of PID-type controller settings via an LQR approach for a two-wheeled balancing robot," 25th International Conference on Methods and Models in Automation and Robotics (MMAR), 2021. DOI: 10.1109/MMAR49549.2021.9528471
[8] V. Klemm et al., "Ascento: A two-wheeled jumping robot," in Proc. IEEE International Conference on Robotics and Automation, 2019, pp. 7515-7521. https://doi.org/10.1109/ICRA.2019.8793792
[9] V. Klemm, A. Morra, L. Gulich, et al., "LQR-assisted whole-body control of a wheeled bipedal robot with kinematic loops," IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 3745-3752, 2020. https://doi.org/10.1109/LRA.2020.2979625
[10] T. Dinev, S. Xin, W. Merkt, V. Ivan, and S. Vijayakumar, "Modeling and control of a hybrid wheeled jumping robot," in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2020. https://doi.org/10.1109/IROS45743.2020.9341339
[11] H. Chen, B. Wang, Z. Hong, C. Shen, P. M. Wensing, and W. Zhang, "Underactuated motion planning and control for jumping with wheeled-bipedal robots," arXiv:2012.06156, 2020. https://arxiv.org/abs/2012.06156
[12] D. Baek, A. Purushottam, and J. Ramos, "Hybrid LMC: Hybrid learning and model-based control for wheeled humanoid robot via ensemble deep reinforcement learning," arXiv:2204.03159, 2022. https://arxiv.org/abs/2204.03159
[13] H. Zhao, L. Yu, S. Qin, G. Jin, and Y. Chen, "Design and control of a bio-inspired wheeled bipedal robot," IEEE/ASME Transactions on Mechatronics, 2024. https://doi.org/10.1109/TMECH.2024.3449397
[14] T. Bishop, K. Ye, and K. Karydis, "Design and central pattern generator control of a new transformable wheel-legged robot," arXiv:2407.03765, 2024. https://arxiv.org/abs/2407.03765
[15] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," Automatica, vol. 36, no. 6, pp. 789-814, 2000. https://doi.org/10.1016/S0005-1098(99)00214-9
[16] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," Annual Review of Control, Robotics, and Autonomous Systems, vol. 3, pp. 269-296, 2020. https://doi.org/10.1146/annurev-control-090419-075625
[17] L. Brunke et al., "Safe learning in robotics: From learning-based control to safe reinforcement learning," Annual Review of Control, Robotics, and Autonomous Systems, vol. 5, pp. 411-444, 2022. https://doi.org/10.1146/annurev-control-042920-020211
[18] K. P. Wabersich and M. N. Zeilinger, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems," 2018. https://arxiv.org/abs/1812.05506
[19] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," International Journal of Robotics Research, vol. 32, no. 11, pp. 1238-1274, 2013. https://doi.org/10.1177/0278364913495721
[20] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, pp. 529-533, 2015. https://doi.org/10.1038/nature14236
[21] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015. https://arxiv.org/abs/1509.02971
[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. https://arxiv.org/abs/1707.06347
[23] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," Proceedings of Machine Learning Research, vol. 80, pp. 1861-1870, 2018. https://proceedings.mlr.press/v80/haarnoja18b.html
[24] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in Proc. IEEE ICRA, 2018, pp. 7559-7566. https://doi.org/10.1109/ICRA.2018.8463169
[25] G. Williams, N. Wagener, B. Goldfain, et al., "Information theoretic MPC for model-based reinforcement learning," in Proc. IEEE ICRA, 2017, pp. 1714-1721. https://doi.org/10.1109/ICRA.2017.7989202
[26] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," Journal of Machine Learning Research, vol. 17, no. 39, pp. 1-40, 2016. https://jmlr.org/papers/v17/15-522.html
[27] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," IEEE Transactions on Robotics and Automation, vol. 8, no. 3, pp. 313-326, 1992. https://doi.org/10.1109/70.143350
[28] 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. https://doi.org/10.1109/70.538972
[29] F. Chaumette and S. Hutchinson, "Visual servo control, Part I: Basic approaches," IEEE Robotics & Automation Magazine, vol. 13, no. 4, pp. 82-90, 2006. https://doi.org/10.1109/MRA.2006.250573
[30] F. Chaumette and S. Hutchinson, "Visual servo control, Part II: Advanced approaches," IEEE Robotics & Automation Magazine, vol. 14, no. 1, pp. 109-118, 2007. https://doi.org/10.1109/MRA.2007.339609
[31] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: A generic software platform with a wide class of robot control skills," IEEE Robotics & Automation Magazine, vol. 12, no. 4, pp. 40-52, 2005. https://doi.org/10.1109/MRA.2005.1577023
[32] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Visual servoing from deep neural networks," 2017. https://arxiv.org/abs/1705.08940
[33] S. Asayesh, H. S. Darani, M. Chen, M. Mehrandezh, and K. Gupta, "Toward scalable visual servoing using deep reinforcement learning and optimal control," arXiv:2310.01360, 2023. https://arxiv.org/abs/2310.01360
[34] L. Zhang, J. Pei, K. Bai, Z. Chen, and J. Zhang, "A closed-loop multi-perspective visual servoing approach with reinforcement learning," in Proc. IEEE International Conference on Robotics and Biomimetics, 2023. https://doi.org/10.1109/ROBIO58561.2023.10354958
[35] F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine, "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control," 2018. https://arxiv.org/abs/1812.00568
[36] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 2017. https://doi.org/10.1109/TPAMI.2016.2577031
[37] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proc. IEEE CVPR, 2016, pp. 779-788. https://doi.org/10.1109/CVPR.2016.91
[38] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in Proc. IEEE CVPR, 2017, pp. 7263-7271. https://doi.org/10.1109/CVPR.2017.690
[39] W. Liu et al., "SSD: Single shot MultiBox detector," in Computer Vision - ECCV 2016, pp. 21-37. https://doi.org/10.1007/978-3-319-46448-0_2

[40] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in Proc. IEEE ICCV, 2017, pp. 2999-3007. https://doi.org/10.1109/ICCV.2017.324
[41] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in Proc. IEEE/CVF CVPR, 2020, pp. 10781-10790. https://doi.org/10.1109/CVPR42600.2020.01079
[42] N. Carion et al., "End-to-end object detection with transformers," in Computer Vision - ECCV 2020, pp. 213-229. https://doi.org/10.1007/978-3-030-58452-8_13
[43] X. Zhu et al., "Deformable DETR: Deformable transformers for end-to-end object detection," 2020. https://arxiv.org/abs/2010.04159
[44] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO series in 2021," 2021. https://arxiv.org/abs/2107.08430
[45] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," 2022. https://arxiv.org/abs/2207.02696
[46] Y. Zhao et al., "DETRs beat YOLOs on real-time object detection," 2023. https://arxiv.org/abs/2304.08069
[47] R. Sapkota et al., "YOLO advances to its genesis: A decadal and comprehensive review of the You Only Look Once series," 2024. https://arxiv.org/abs/2406.19407
[48] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in Proc. IEEE ICIP, 2017, pp. 3645-3649. https://doi.org/10.1109/ICIP.2017.8296962
[49] P. Bergmann, T. Meinhardt, and L. Leal-Taixé, "Tracking without bells and whistles," in Proc. IEEE/CVF ICCV, 2019, pp. 941-951. https://doi.org/10.1109/ICCV.2019.00103
[50] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "FairMOT: On the fairness of detection and re-identification in multiple object tracking," International Journal of Computer Vision, vol. 129, pp. 3069-3087, 2021. https://doi.org/10.1007/s11263-021-01513-4
[51] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," in Computer Vision - ECCV 2020, pp. 474-490. https://doi.org/10.1007/978-3-030-58548-8_28
[52] Y. Zhang et al., "ByteTrack: Multi-object tracking by associating every detection box," in Computer Vision - ECCV 2022, pp. 1-21. https://doi.org/10.1007/978-3-031-20047-2_1
[53] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, "BoT-SORT: Robust associations multi-pedestrian tracking," 2022. https://arxiv.org/abs/2206.14651
[54] S. Y. Chen, "Kalman filter for robot vision: A survey," IEEE Transactions on Industrial Electronics, vol. 59, no. 11, pp. 4409-4420, 2012. https://doi.org/10.1109/TIE.2011.2162714
[55] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in Proc. IEEE ICRA, 2007, pp. 3565-3572. https://doi.org/10.1109/ROBOT.2007.364024
[56] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," International Journal of Robotics Research, vol. 34, no. 3, pp. 314-334, 2015. https://doi.org/10.1177/0278364914554813
[57] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," IEEE Transactions on Robotics, vol. 33, no. 5, pp. 1255-1262, 2017. https://doi.org/10.1109/TRO.2017.2705103
[58] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," IEEE Transactions on Robotics, vol. 34, no. 4, pp. 1004-1020, 2018. https://doi.org/10.1109/TRO.2018.2853729
[59] C. Campos et al., "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," IEEE Transactions on Robotics, vol. 37, no. 6, pp. 1874-1890, 2021. https://doi.org/10.1109/TRO.2021.3075644
[60] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," in Proc. IEEE/RSJ IROS, 2020, pp. 5135-5142. https://doi.org/10.1109/IROS45743.2020.9341176
[61] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in Proc. IEEE/CVF CVPR, 2021. https://arxiv.org/abs/2104.09224
[62] Z. Liu, H. Tang, A. Amini, X. Yang, H. Mao, D. Rus, and S. Han, "BEVFusion: Multi-task multi-sensor fusion with unified bird's-eye view representation," in Proc. IEEE International Conference on Robotics and Automation, 2023. https://arxiv.org/abs/2205.13542
[63] F. Huang, S. Liu, G. Zhang, et al., "DeployFusion: A deployable monocular 3D object detection with multi-sensor information fusion in BEV for edge devices," Sensors, vol. 24, no. 21, art. 7007, 2024. https://doi.org/10.3390/s24217007
[64] B. Blanco-Filgueira, D. García-Lesta, M. Fernández-Sanjurjo, V. M. Brea, and P. López, "Deep learning-based multiple object visual tracking on embedded system for IoT and mobile edge computing applications," arXiv:1808.01356, 2018. https://arxiv.org/abs/1808.01356
[65] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF CVPR, 2018, pp. 4510-4520. https://doi.org/10.1109/CVPR.2018.00474
[66] B. Jacob et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in Proc. IEEE/CVF CVPR, 2018, pp. 2704-2713. https://doi.org/10.1109/CVPR.2018.00286
[67] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015. https://arxiv.org/abs/1510.00149
[68] A. Dmytruk, G. Silano, D. Bicego, D. Bonilla Licea, and M. Saska, "A perception-aware NMPC for vision-based target tracking and collision avoidance with a multi-rotor UAV," in Proc. International Conference on Unmanned Aircraft Systems, 2022, pp. 1668-1673. https://doi.org/10.1109/ICUAS54217.2022.9836071
[69] J. P. K. V. Pothuri, A. Bhatt, P. KrisshnaKumar, M. Oddiraju, and S. Chowdhury, "Learning approach to efficient vision-based active tracking of a flying target by an unmanned aerial vehicle," arXiv:2506.18264, 2025. https://arxiv.org/abs/2506.18264
[70] W. Luo, P. Sun, F. Zhong, W. Liu, T. Zhang, and Y. Wang, "End-to-end active object tracking and its real-world deployment via reinforcement learning," 2018. https://arxiv.org/abs/1808.03405