REVIEW ARTICLE

A Review on Embedded Arduino-Based Visual Robotic Arm Grasping and SLAM Navigation

Fajun Guo[1,*]

[1] College of Engineering, Lishui University, Lishui 323000, China

Abstract

With the development of desktop robotic arms towards low cost, lightweight and intelligence, the embedded Arduino vision-based robotic arm grasping technology has become a research hotspot. This paper reviews the current research status of the robotic arm vision grasping technology and embedded SLAM navigation with Arduino as the lower computer. Firstly, the embedded Arduino vision-based robotic arm grasping is introduced, and the system architecture, mainstream kinematic modeling methods, visual calibration, pose estimation, dynamic analysis and stability, and trajectory planning processes are sorted out in sequence. Then, the shortcomings of the system in terms of adaptability to complex environments, dynamic response and calibration dependence are pointed out. In addition, the application of lightweight embedded SLAM is introduced. Finally, the article is summarized and prospected, which can provide a reference for the research on the intelligence of small robotic arms.

Keywords: arduino, embedded SLAM, visual grasping, robotic arm control.

Submitted: 10 May 2026
Accepted: 27 May 2026
Published: 07 June 2026

Vol. 1, **No.** 2, 2026.
10.62762/TICPS.2026.578769

***Corresponding author:**
Fajun Guo
1519855759@qq.com

1 Introduction

specialized equipment into multi-capability general-purpose devices. Representative efforts in control system optimization, such as the application of fuzzy PID algorithms to six-degree-of-freedom configurations [1], reflect the broader trend of expanding their application scope to agriculture, service industries, and the aerospace sector. In particular, with the successive advent of uArm and Magician in the 2010s, low-cost, lightweight, and compact desktop robotic arms have emerged as a research hotspot in recent years. To date, research on vision-based intelligent grasping for Arduino-based robotic arms has become relatively mature. Researchers began to attempt to introduce the YOLO algorithm [2] or the TensorFlow framework [3] for further optimization. As robotic systems grow in complexity-encompassing kinematic modeling, sensor integration, and autonomous control as comprehensively treated in foundational references [4]-the integration of SLAM (Simultaneous Localization and Mapping) technology into fixed-base robotic arms constitutes an increasingly important research focus and challenge in this domain. This paper primarily summarizes the visual grasping methods for robotic arms using Arduino as the lower-level controller—covering system architecture, kinematic modeling, visual calibration, pose estimation, dynamic analysis and stability, and trajectory planning—as well as the application of

Citation
Guo, F. (2026). A Review on Embedded Arduino-Based Visual Robotic Arm Grasping and SLAM Navigation. *ICCK Transactions on Intelligent Cyber-Physical Systems*, 1(2), 60–65.

SLAM navigation algorithms in embedded systems, thereby providing a comprehensive reference for research in relevant fields.

2 Research Status of Vision-Based Grasping Technology for Embedded Arduino Robotic Arms

The visual grasping process of a robotic arm typically follows the following workflow: First, the system architecture is defined; second, kinematic analysis is performed based on the robotic arm's structural parameters; third, visual calibration and pose estimation are sequentially implemented; and finally, trajectory planning is completed to execute the grasping operation.

2.1 An Introduction to the Architecture of Robotic Arm Systems and Arduino-Based Architectures

Lightweight small-scale desktop robotic arm systems typically employ a hierarchical upper-lower computer control architecture. The lower computer, primarily based on Arduino [5] (with some implementations using STM32, a microcontroller widely adopted in embedded motion control applications [6]), utilizes a microcontroller to execute low-level logic control. The upper computer, predominantly a PC [7], operates within environments such as MATLAB, Python [9], and LabVIEW [10]. Alternatively, embedded development boards like Raspberry Pi [11] and Jetson Nano—whose on-board inference capability enables real-time perception tasks such as object detection and inventory analysis [12]—are sometimes adopted as upper computers for autonomous computing.

Overall, the Arduino-PC architecture is the most widely adopted. Sun and Bei [7] employed MATLAB for image acquisition and analysis, and developed a GUI interface to enable communication with Arduino for motor rotation control. Mao et al. [8] performed visual image analysis via MATLAB and realized accurate recognition, positioning, and grasping of workpieces through end-effector control of the manipulator.

2.2 Kinematic Modeling and Analysis for Robotic Arms

Kinematic analysis is further categorized into forward kinematics and inverse kinematics. Robot forward kinematics refers to the calculation of the end effector's pose based on the known parameters of each link and joint, while inverse kinematics addresses the inverse problem of the forward kinematic solution [13]. Prior to conducting kinematic analysis, however, it is essential to establish the geometric relationships between the links of the robotic arm. A variety of methods are currently available, with the following being the mainstream kinematic modeling approaches:

2.2.1 D-H(Denavit-Hartenberg) Method

The standard Denavit-Hartenberg (D-H) method constructs an attached coordinate system for each link and defines four geometric parameters (link length, twist angle, joint offset, and joint angle), thereby representing the pose transformation between adjacent links as the successive multiplication of homogeneous transformation matrices. While this method features intuitive geometric meaning and computational simplicity, it is plagued by singularity issues. In contrast, the modified D-H method introduced by Craig in his monograph [14] exhibits a more explicit physical-geometric correspondence. The core improvements of the modified D-H method lie in adjusting the attachment position of the coordinate system (shifting from the distal end to the proximal end of the link) and reordering the transformation sequence. Presently, the modified D-H method has emerged as the mainstream kinematic modeling approach and is widely applied across various fields. Researchers such as Chen et al. [15] and Song et al. [16] have all conducted their studies based on kinematic modeling using the D-H method.

2.2.2 Geometric vector Method

The geometric vector method is essentially an optimization of the traditional Denavit-Hartenberg (D-H) method. While the traditional D-H method treats link length as a scalar, the geometric vector method redefines link length as a vector, enabling more precise description of the pose of complex links. However, this method has not yet established a unified standardized representation, and its application in practical engineering remains relatively limited. Chanal et al. [17] verified the effectiveness of the geometric vector method in the modeling of multi-degree-of-freedom (DOF) robots. Pikalov et al. [18] proposed a vector-model-based kinematic representation for six rotational joint robots.

2.2.3 Product of Exponentials (POE) Method

The product of exponentials (POE) method, rooted in screw theory, employs a globally fixed base coordinate system and tool coordinate system. It abstracts the motion of each joint as a helical motion about a spatially fixed axis, and directly derives the end-effector pose via the product of matrix exponentials. This method circumvents the construction and successive transformation of intermediate coordinate systems, thereby naturally eliminating singularity issues. However its inverse kinematics solution remains highly complex. Du and Chen [19] solved the inverse solution using a locally chaotic particle swarm optimization algorithm; Malik et al. [20] obtained the inverse solution through swarm intelligence (SI).

Most desktop robotic arms adopt the Denavit-Hartenberg (D-H) method or its modified variant. This is primarily because MATLAB, which serves as the upper computer environment in many implementations, provides convenient support for kinematic modeling and trajectory computation, as demonstrated in Cartesian-space planning studies [21], thereby obviating the tediousness of traditional manual calculations.

2.3 Visual Calibration

To achieve visual calibration, an accurate geometric mapping relationship must be established among the pixel coordinate system, camera coordinate system, world coordinate system, and robot end-effector coordinate system. This process first relies on camera calibration to realize the projection mapping from 3D space points to 2D image points, followed by hand-eye calibration to solve for the fixed rigid-body transformation between the camera coordinate system and the robot end-effector coordinate system.

Camera calibration mainly relies on Zhang's calibration method to obtain the optimal intrinsic parameters, distortion coefficients, and extrinsic parameters. Hand-eye calibration is divided into two categories according to the camera installation position: eye-in-hand and eye-to-hand [22]. The eye-in-hand configuration means the camera is rigidly mounted on the end-effector of the manipulator. Its field of view moves synchronously with the manipulator, featuring no blind zone and flexible workpiece grasping, but it is prone to vibration. The eye-to-hand configuration means the camera is fixed on the environmental base. The camera is stable with high solution accuracy, but has a blind zone in the field of view. In addition, cameras are classified into monocular [23] and binocular types according to the number of cameras. A monocular camera is low-cost but has limited accuracy and cannot acquire depth information, while a binocular camera provides depth perception and higher spatial resolution at the cost of increased system complexity—a configuration exploited, for instance, in brain-machine interface-driven assistive grasping systems where reliable depth estimation is essential [24].

2.4 Pose Estimation

Pose estimation involves solving for the pose of a target object using the calibration parameters obtained earlier. Based on the technological development trajectory, pose estimation methods can be broadly categorized into traditional geometric methods and deep learning-based methods. Traditional geometric methods rely on artificially designed geometric features and constraints to compute the pose between the object and the camera. For instance, Cui et al. [25] and Zheng et al. [23] both employed the Perspective-n-Point (PnP) algorithm for pose estimation. Deep learning-based methods leverage deep learning networks to automatically learn features. Zhang et al. [26], for example, utilized point clouds for pose estimation.

2.5 Dynamic Analysis and Stability

Dynamic analysis aims to investigate the dynamic response and stability of a system subjected to external forces and torques. According to technical frameworks, existing approaches are categorized into two groups: traditional physics-model-based methods and data-driven intelligent methods. Conventional methods [27] establish explicit dynamic models via Lagrange formulation or Newton–Euler algorithm; in contrast, intelligent control methods based on adaptive and robust frameworks employ online approximation strategies—such as terminal sliding mode schemes—to compensate for unknown system nonlinearities without requiring precise dynamic parameters [28].

After dynamic modeling, stability analysis is required to verify the convergence performance of the designed controllers. For traditional modeling schemes with accurate analytical dynamics, the classical Lyapunov direct method combined with LaSalle's invariance principle is widely adopted for stability proof [29]. Intelligent control approaches mostly rely on the Lyapunov framework for stability verification, including Lyapunov synthesis for uniformly ultimately bounded (UUB) derivation [28] and finite-time Lyapunov criterion [30].

2.6 Trajectory Planning

Trajectory planning involves designing a safe, efficient, and smooth execution path based on the output of pose estimation. Arduino-based systems typically employ two primary trajectory planning approaches: joint space trajectory planning and Cartesian space trajectory planning.

Joint space planning directly interpolates the angles of individual joints, with polynomial interpolation being the most representative method. For example, Fang et al. [31] adopted quintic polynomial trajectory planning for a seven-DOF manipulator, demonstrating its applicability to higher-dimensional configurations. Xu et al. [32] combined quintic polynomial interpolation with an improved Harris Hawks optimization algorithm to achieve time-optimal planning, which ensures the continuity of angular acceleration and thus enhances motion smoothness.

Cartesian space planning focuses on the spatial path of the end-effector, where linear interpolation and circular interpolation are the fundamental forms. Huang et al. [33] designed a linear interpolation algorithm to guarantee the smoothness of Cartesian space trajectory planning. Zhang et al. [34] applied circular arc interpolation trajectory planning in a LabVIEW-based manipulator system, demonstrating smooth end-effector motion in Cartesian space. Wei et al. [35] comprehensively employed linear and circular interpolation to complete the simulation of rectangular and elliptical trajectories in MATLAB.

2.7 Limitations of Visual-Grasping Fusion

Despite the widespread application of Arduino-based fixed robotic arm visual grasping technology in cost-sensitive scenarios, the inherent performance constraints of the Arduino platform result in several notable limitations. Firstly, the system demonstrates inadequate adaptability to complex environments, as the limited imaging accuracy of low-cost vision modules often induces pose estimation deviations and subsequent grasping failures. Secondly, the approach exhibits a heavy reliance on calibration accuracy; since the platform lacks the computational capacity to support complex optimization algorithms, the accumulation of calibration errors inevitably degrades grasping positioning precision. Thirdly, dynamic response is often lagging, where the restricted computational speed of the controller, combined with the inherent low flexibility of fixed manipulators, frequently leads to trajectory deviations. Furthermore, the limited payload capacity of Arduino-controlled robotic arms restricts their applicability in tasks requiring the manipulation of medium-to-large workpieces.

3 Embedded Simultaneous Localization and Mapping (SLAM) Technology

SLAM navigation enables autonomous localization and mapping in unknown environments, thereby providing target-point navigation capabilities for robotic arm grasping. However, the Arduino platform—characterized by limited computing power and small memory—cannot directly run traditional SLAM algorithms. Currently, mainstream embedded SLAM navigation solutions are dominated by visual SLAM and visual-inertial SLAM.

Giubilato et al. [36] proposed a visual-inertial SLAM algorithm tailored for embedded platforms, which was successfully deployed on the NVIDIA Jetson TX2 embedded computing platform. Malakouti-Khah et al. [37] designed and implemented a lightweight heterogeneous multi-robot cooperative SLAM system, and achieved successful deployment on multiple Raspberry Pi-based mobile robots. Luo et al. [38] improved the ORB-SLAM visual odometry and applied the optimized version to the ZYNQ embedded system.

4 Conclusion

This paper reviews the current research status of visual grasping technology for Arduino-based fixed-base robotic arms and embedded SLAM navigation. Existing findings indicate that the hierarchical architecture, where Arduino serves as the lower-level controller paired with an upper-level computer, has been maturely applied in low-cost desktop robotic arms. However, constrained by the limited computing power and memory of the Arduino platform, the system still exhibits distinct limitations in complex environment adaptability, calibration accuracy, and dynamic response. Concurrently, the advancement of embedded SLAM provides a feasible solution for robotic arms to achieve autonomous localization, map construction, and navigation in unknown environments.

Future research can pursue breakthroughs in the following directions: First, implement an integrated design that unifies lightweight visual SLAM navigation with robotic arm grasping tasks; second, develop ultra-lightweight SLAM algorithms tailored specifically for the Arduino platform.

Data Availability Statement

Not applicable.

Funding

This work was supported without any funding.

Conflicts of Interest

The author declares no conflicts of interest.

AI Use Statement

The author declares that no generative AI was used in the preparation of this manuscript.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Wang, C., & Jiang, J. (2024, July). Optimization Design of Six-Degree-of-Freedom Robot Arm Control System Based on Fuzzy Pid Algorithm. In *2024 2nd International Conference on Algorithm, Image Processing and Machine Vision (AIPMV)* (pp. 99-104). IEEE. [CrossRef]

[2] Panigrahi, S., Biyani, A., Mithari, A., Sawant, S., Tade, S., & Kolte, M. (2023, August). YOLO on GPU Implementation for a Vision-Based Fruit Select and Place System with a 5-DOF Robotic Arm. In *2023 7th International Conference On Computing, Communication, Control And Automation (ICCUBEA)* (pp. 1-7). IEEE. [CrossRef]

[3] Nandan, N., & Thippeswamy, K. (2018, December). A tensorflow based robotic arm. In *2018 international conference on electrical, electronics, communication, computer, and optimization techniques (ICEECCOT)* (pp. 326-330). IEEE. [CrossRef]

[4] Spong, M. W., Hutchinson, S., & Vidyasagar, M. (2020). *Robot modeling and control* (Vol. 2). New York: Wiley. [CrossRef]

[5] Mohammed Ali, H., Hashim, Y., & A AL-Sakkal, G. (2022). Design and implementation of Arduino based robotic arm. *International Journal of Electrical and Computer Engineering, 12*(2), 1411-1411. [CrossRef]

[6] Long, T. (2023, March). Design of sweeping robot based on stm32 single chip microcomputer. In *Journal of Physics: Conference Series* (Vol. 2456, No. 1, p. 012045). IOP Publishing. [CrossRef]

[7] Sun, G., & Bei, G. (2023). Arduino-based intelligent handling robot design. *Advances in Computer, 496*, 67-74. [CrossRef]

[8] Mao, Y., Chen, C., & Jiang, H. (2022, March). Design and implementation of sorting system based on machine vision. In *2022 7th International Conference on Big Data Analytics (ICBDA)* (pp. 259-264). IEEE. [CrossRef]

[9] Lee, H. W. (2020). The study of mechanical arm and intelligent robot. *IEEE Access, 8*, 119624-119634. [CrossRef]

[10] Mouli, C. C., Jyothi, P., Raju, K. N., & Nagaraja, C. (2013). Design and implementation of robot arm control using labview and arm controller. *IOSR Journal of Electrical and Electronics Engineering, 6*(5), 80-84. [CrossRef]

[11] Pereira, V., Fernandes, V. A., & Sequeira, J. (2014, September). Low cost object sorting robotic arm using Raspberry Pi. In *2014 IEEE global humanitarian technology conference-South Asia Satellite (GHTC-SAS)* (pp. 1-6). IEEE. [CrossRef]

[12] Bharadi, V., Mukadam, S., Prasad, R., Upparakakula, K., & Jaygade, J. (2023). Real-time inventory analysis using Jetson Nano with object detection and analysis. In *Deep Learning-Recent Findings and Research. IntechOpen*. [CrossRef]

[13] Liu, B. B., Yuan, L., Kong, Q. B., & Wu, J. Q. (2020). Research on Motion Trajectory Optimization Method of 6-DOF Industrial Robot. *Modular Machine Tool & Automatic Manufacturing Technique*, (2), 11–15. [CrossRef]

[14] Craig, J. J. (2009). *Introduction to robotics: mechanics and control, 3/E*. Pearson Education India.

[15] Chen, D. G., Liu, X. D., Qian, C., & Kong, Z. C. (2024). Improved Particle Swarm Optimization Trajectory Optimization Planning Algorithm for Industrial Robotic Arms. *Manufacturing Automation, 46*(11), 57–63. [CrossRef]

[16] Song, Q., Li, S., Bai, Q., Yang, J., Zhang, A., Zhang, X., & Zhe, L. (2021). Trajectory planning of robot manipulator based on RBF neural network. *Entropy, 23*(9), 1207. [CrossRef]

[17] Chanal, H., Guyon, J. B., Koessler, A., Dechambre, Q., Boudon, B., Blaysat, B., & Bouton, N. (2021). Geometrical defect identification of a SCARA robot from a vector modeling of kinematic joints invariants. *Mechanism and Machine Theory, 162*, 104339. [CrossRef]

[18] Pikalov, I., Spirin, E., Saramud, M., & Kubrikov, M. (2022). Vector model for solving the inverse kinematics problem in the system of external adaptive control of robotic manipulators. *Mechanism and Machine Theory, 174*, 104912. [CrossRef]

[19] Du, Y., & Chen, Y. (2022). Time optimal trajectory planning algorithm for robotic manipulator based on locally chaotic particle swarm optimization. *Chinese Journal of Electronics, 31*(5), 906-914. [CrossRef]

[20] Malik, A., Henderson, T., & Prazenica, R. (2021). Multi-objective swarm intelligence trajectory generation for a 7 degree of freedom robotic manipulator. *Robotics, 10*(4), 127. [CrossRef]

[21] Qin, Q., Guo, Y., & Dynav. (2022, September). Trajectory Planning of Cartesian Coordinate Robot Based on Combinatorial Optimization Algorithm. In *International Conference on Cognitive based Information*

Processing and Applications (pp. 437-445). Singapore: Springer Nature Singapore. [CrossRef]

[22] Enebuse, I., Foo, M., Ibrahim, B. S. K. K., Ahmed, H., Supmak, F., & Eyobu, O. S. (2021). A comparative review of hand-eye calibration techniques for vision guided robots. *IEEE Access, 9*, 113143-113155. [CrossRef]

[23] Zheng, Z., Ma, Y., Zheng, H., Gu, Y., & Lin, M. (2018). Industrial part localization and grasping using a robotic arm guided by 2D monocular vision. *Industrial Robot: An International Journal, 45*(6), 794-804. [CrossRef]

[24] Shim, K. H., Jeong, J. H., Kwon, B. H., Lee, B. H., & Lee, S. W. (2019, October). Assistive robotic arm control based on brain-machine interface with vision guidance using convolution neural network. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)* (pp. 2785-2790). IEEE. [CrossRef]

[25] Cui, X., Yu, M., Wu, L., & Wu, S. (2022). A 6D pose estimation for robotic bin-picking using point-pair features with curvature (Cur-PPF). *Sensors, 22*(5), 1805. [CrossRef]

[26] Zhang, Q., Xue, C., Qin, J., Duan, J., & Zhou, Y. (2024). 6d pose estimation of industrial parts based on point cloud geometric information prediction for robotic grasping. *Entropy, 26*(12), 1022. [CrossRef]

[27] Siciliano, B., Sciavicco, L., Villani, L., & Oriolo, G. (2009). *Robotics: modelling, planning and control*. London: Springer London. [CrossRef]

[28] Chen, M., Wu, Q. X., & Cui, R. X. (2013). Terminal sliding mode tracking control for a class of SISO uncertain nonlinear systems. *ISA transactions, 52*(2), 198-206. [CrossRef]

[29] Xu, F., Tang, L., & Liu, Y. J. (2021). Tangent barrier Lyapunov function-based constrained control of flexible manipulator system with actuator failure. *International Journal of Robust and Nonlinear Control, 31*(17), 8523-8536. [CrossRef]

[30] Yu, S., Yu, X., Shirinzadeh, B., & Man, Z. (2005). Continuous finite-time control for robotic manipulators with terminal sliding mode. *Automatica, 41*(11), 1957-1964. [CrossRef]

[31] Fang, S., Ma, X., Zhao, Y., Zhang, Q., & Li, Y. (2019, August). Trajectory planning for seven-DOF robotic arm based on quintic polynomial. In *2019 11th international conference on intelligent human-machine systems and cybernetics (IHMSC)* (Vol. 2, pp. 198-201). IEEE. [CrossRef]

[32] Xu, J., Ren, C., & Chang, X. (2023). Robot time-optimal trajectory planning based on quintic polynomial interpolation and improved Harris Hawks algorithm. *Axioms, 12*(3), 245. [CrossRef]

[33] Huang, J., Xiao, S., Wang, Z. Q., & Wang, X. L. (2019, October). Research on Trajectory Planning Algorithms of Cartesian Space for Robots. In *2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)* (pp. 1204-1208). IEEE. [CrossRef]

[34] Zhang, W., Ma, J., Ye, Y., Zhou, S., Ye, X., & You, Z. (2022, October). Circular Interpolation Trajectory Control of Manipulator Based on LabVIEW. In *2022 4th International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)* (pp. 794-801). IEEE. [CrossRef]

[35] Wei, B., Liu, C., Zhang, X., Zheng, K., Cao, Z., & Chen, Z. (2024). Smooth and time-optimal trajectory planning for robots using improved carnivorous plant algorithm. *Machines, 12*(11), 802. [CrossRef]

[36] Giubilato, R., Chiodini, S., Pertile, M., & Debei, S. (2019). An evaluation of ROS-compatible stereo visual SLAM methods on a nVidia Jetson TX2. *Measurement, 140*, 161-170. [CrossRef]

[37] Malakouti-Khah, H., Sadeghzadeh-Nokhodberiz, N., & Montazeri, A. (2024). Simultaneous localization and mapping in a multi-robot system in a dynamic environment with unknown initial correspondence. *Frontiers in Robotics and AI, 10*, 1291672. [CrossRef]

[38] Luo, Q., Zhu, J., & Pei, H. (2024, November). ORB-SLAM3 Front-End Acceleration System Based on ZYNQ Platform. In *2024 China Automation Congress (CAC)* (pp. 2203-2208). IEEE. [CrossRef]

Fajun Guo received his bachelor's degree from the University of Science and Technology Liaoning and is currently pursuing a master's degree in Mechanical Engineering at the Graduate School of Lishui University. His research focuses on core algorithms for modern industrial robotic arms, with particular emphasis on enhancing the overall motion accuracy and intelligent grasping capabilities of small desktop manipulators. (Email: 1519855759@qq.com)