INTERNATIONAL JOURNAL OF MODERN COMPUTER SCIENCE AND IT INNOVATIONS (IJMCSIT)

eISSN: 3087-4289

Volume. 02, Issue. 01, pp. 08-15, January 2025"

AFFORDABLE VISION-BASED SYSTEMS FOR REAL-TIME CHESSBOARD DIGITIZATION

Dr. Ahmed R. Mostafa
Department of Computer Systems and Robotics, Alexandria University, Egypt

Prof. Mahmoud A. Taha
Department of Computer Engineering, Alexandria University, Egypt

Article received: 08/10/2024, Article Revised: 13/12/2024, Article Accepted: 11/01/2025
DOI: https://doi.org/10.55640/ijmcsit-v02i01-02

ABSTRACT

The automatic recognition of chessboard states has significant applications in various domains, from enhancing online chess platforms and educational tools to enabling robotic interaction. While high-performance vision systems and complex robotic setups can achieve this, their cost and complexity often limit widespread adoption. This paper explores the feasibility and methodology for developing affordable, vision-based systems for real-time chessboard digitization. We leverage advancements in deep learning, particularly lightweight Convolutional Neural Networks (CNNs), combined with accessible embedded platforms. The proposed approach integrates image acquisition, chessboard localization, and individual chess piece recognition, culminating in a standardized digital representation of the board state. Our findings demonstrate that acceptable levels of accuracy and real-time performance can be achieved on low-cost hardware, making automatic chess digitization more accessible for a broader range of applications.

Keywords: Chessboard digitization, real-time processing, vision-based systems, affordable hardware, computer vision, image recognition, board game analysis, object detection, pattern recognition, low-cost AI solutions.

INTRODUCTION

Chess, a game of strategy and intellect, has captivated humanity for centuries. In the modern era, the digital realm has profoundly influenced how chess is played, studied, and analyzed. Automatic digitization of the chessboard state – the ability to automatically identify the position and type of all pieces on a physical board – forms a critical bridge between the physical and digital chess worlds. This capability is invaluable for various applications, including:

- Enhanced Online Play: Allowing players to use physical boards while their moves are automatically transmitted to online platforms.
- Chess Analytics and Training: Providing real-time feedback and analysis for players using traditional boards.
- Robotic Chess Systems: Enabling robots to perceive the board and interact with human players [16, 17, 18, 19].
- Educational Tools: Facilitating interactive learning experiences for beginners.

The challenge of automatically recognizing the state of a chessboard is primarily a computer vision problem. Traditional approaches have often relied on a combination of image processing techniques such as edge detection [48], line detection (e.g., Hough transform) [49], and feature matching [26, 27] to locate the board and distinguish pieces. While these methods can work in controlled environments, their robustness often falters under varying lighting conditions, camera angles, or diverse piece designs.

The advent of deep learning, particularly Convolutional Neural Networks (CNNs) [1, 2, 3, 8], has revolutionized computer vision, offering unprecedented accuracy in image classification and object detection. CNNs, exemplified by architectures like AlexNet [2], ResNet [3], MobileNetV2 [5], and Xception [6], have demonstrated remarkable capabilities in learning complex features directly from raw image data. However, deploying these computationally intensive models on embedded, low-cost hardware presents its own set of challenges regarding processing power, memory footprint, and energy consumption [4, 20, 21, 22, 31].

Despite the advancements, a significant gap remains in developing integrated, affordable solutions for automatic chess digitization that are accessible to hobbyists, educators, and small-scale developers. Existing high-performance systems often rely on powerful computing hardware or specialized sensors, increasing their cost and complexity.

This paper aims to bridge this gap by exploring a methodology for building low-cost, vision-based automatic chess digitization systems. Our core idea is to combine robust computer vision techniques with efficient deep learning models suitable for deployment on affordable embedded platforms. We present a comprehensive approach, from image acquisition and board localization to piece recognition and board state generation, emphasizing practical implementation on resource-constrained devices.

The key contributions of this work are:

- A methodology for robust chessboard localization and piece recognition using a combination of traditional computer vision and lightweight deep learning models suitable for low-cost platforms.
- An investigation into the performance trade-offs of deploying such systems on affordable embedded hardware.
- A demonstration of the feasibility of achieving real-time or near real-time automatic chess digitization without requiring high-end computing resources.

The remainder of this article is organized as follows: Section 2 reviews relevant literature on chess recognition, deep learning, and embedded vision systems. Section 3 details the proposed methodology for chessboard digitization. Section 4 presents our experimental results and performance analysis. Finally, Section 5 discusses the implications, limitations, and future directions of this research.

2. Related Work

The problem of automatic chessboard state recognition is an interdisciplinary challenge drawing from computer vision, robotics, and artificial intelligence. This section provides an overview of existing approaches and relevant advancements.

2.1. Chessboard and Chess Piece Recognition

Early attempts at chessboard recognition often relied on classic image processing techniques. These involved detecting lines using algorithms like the Hough Transform [49] or Canny edge detection [48] to identify the grid, followed by geometrical analysis to determine the board's perspective [23]. Piece recognition in these systems was typically based on feature extraction (e.g., SIFT [26], shape analysis) and template matching [27]. However, these methods struggled with variations in lighting, background, and piece designs, requiring strict environmental controls.

With the rise of deep learning, particularly Convolutional Neural Networks (CNNs) [29], the accuracy and robustness of image-based recognition tasks, including chess, have dramatically improved [8]. Architectures like AlexNet [2], ResNet [3], and more recently, efficient networks such as MobileNetV2 [5], Xception [6], SqueezeNet [33], and DenseNet [32], have become standard for visual classification tasks due to their ability to learn hierarchical features directly from data [4]. These models have been applied to chess piece recognition, where they classify individual pieces or entire board squares. For instance, Czyzewski et al. used neural networks for chessboard and chess piece recognition [9]. Delgado Neto and Campello explored fine-tuning deep neural networks using synthetically generated chess piece images for robust identification [10], a crucial technique given the diversity of physical chess sets. Ding's work also explored chess board and piece recognition using computer vision techniques [11].

2.2. Robotic and Autonomous Chess Systems

Automatic chess digitization is a key component in robotic chess systems. These systems typically integrate vision modules with robotic manipulators to play chess autonomously or collaboratively with humans. Examples include Gambit, an autonomous chess-playing robotic system developed by Matuszek et al. [16], and systems developed by Chen and Wang for robust computer vision chess analysis and interaction with humanoid robots [17]. Kolosowski et al. also presented a collaborative robot system for playing chess [18]. More recently, Tan's Master's thesis explored vision-based mobile manipulators for autonomous chess gameplay [19]. These projects highlight the demand for accurate and real-time board state recognition, often relying on high-performance computing to meet their stringent requirements.

2.3. Deep Learning on Low-Cost Embedded Systems

The deployment of deep learning models on resource-constrained embedded systems is a rapidly growing field [4, 20, 21, 22]. This trend is driven by the need for on-device AI capabilities in applications ranging from driver fatigue detection [20] and human monitoring [21] to road surface recognition [22]. Efficient processing of deep neural networks on such platforms requires careful consideration of model architecture, optimization techniques, and hardware accelerators [4].

Lightweight CNN architectures, like MobileNetV2 [5] and SqueezeNet [33], are specifically designed to reduce computational complexity and memory footprint while maintaining competitive accuracy. Techniques such as quantization, pruning, and neural architecture search (e.g., NASNet [7]) are employed to further optimize models for embedded deployment.

The emergence of open-source RISC-V processors, such as those generated by the Rocket Chip generator [12, 36], has also facilitated the development of custom hardware for deep learning inference on edge devices [37, 38]. Projects like Gemmini [15] provide systematic deep-learning architecture evaluation via full-stack integration, while others focus on enhancing energy efficiency through techniques like frame buffer compression [14] or hardware-software co-designed accelerators [37]. This integration of custom hardware with optimized software is crucial for achieving high performance on low-cost platforms.

Our work builds upon these advancements by combining established computer vision techniques for board localization with efficient CNN models, optimized for deployment on affordable embedded hardware, to enable practical and accessible automatic chess digitization.

3. Methodology: Vision-Based Chessboard Digitization

Our methodology for automatic chess digitization on low-cost platforms comprises several key stages: image acquisition, chessboard localization, chess piece recognition, and board state encoding. Each stage is designed with computational efficiency and deployment on resource-constrained embedded systems in mind.

3.1. System Architecture

The proposed system utilizes a simple, low-cost hardware setup. The core components include:

- Camera Module: A standard USB camera or a Raspberry Pi Camera Module, chosen for its affordability and ease of integration. The camera is positioned overhead, looking down at the chessboard to simplify perspective handling.
- Embedded Processing Unit: A low-cost single-board computer (SBC) such as a Raspberry Pi (e.g., Raspberry Pi 4) or a RISC-V development board (e.g., HiFive Unmatched, or boards incorporating specialized accelerators like Gemmini [15]). These platforms offer a balance of processing power, I/O capabilities, and energy efficiency suitable for on-device inference [20, 21, 22].
- Chessboard: A standard physical chessboard with consistent square colors and distinct piece designs.

The system workflow is as follows: the camera captures an image of the board; the embedded processor runs the vision algorithms to locate the board and identify pieces; finally, the digitized board state is output, typically as a Forsyth-Edwards Notation (FEN) string [28].

3.2. Image Acquisition and Pre-processing

Images are captured from a fixed, top-down perspective to minimize perspective distortion and simplify subsequent processing. Basic image pre-processing steps are applied:

- Grayscale Conversion: Converting the image to grayscale reduces computational load for initial processing, focusing on structural features.
- Noise Reduction: Applying a Gaussian blur or similar filter to reduce noise and smooth edges.

3.3. Chessboard Localization

Accurate localization of the chessboard within the captured image is paramount. We employ a hybrid approach combining traditional computer vision techniques for robustness and efficiency:

1. Edge Detection: The Canny edge detector [48] is applied to highlight prominent edges in the pre-processed image. This step is computationally inexpensive and effective at revealing the grid lines of the chessboard.

2. Line Detection: The Hough Transform [49] is used to detect straight lines from the Canny edge map. We filter these lines based on their orientation (horizontal and vertical) and length to isolate potential chessboard grid lines.

3. Intersection Detection: The intersections of these detected lines are computed [35]. These intersections correspond to the corners of the chessboard squares. By analyzing the density and spacing of these intersections, the 64 squares of the chessboard can be precisely identified. This stage is critical for establishing a coordinate system for the board.

4. Perspective Correction (if necessary): Although a top-down view is preferred, minor perspective distortions can occur. A homography matrix can be calculated from the detected four outer corners of the

chessboard and applied to warp the board region into a perfectly frontal view, ensuring consistent input for piece recognition [23].

3.4. Chess Piece Recognition with Lightweight CNNs

Once the chessboard is localized and potentially rectified, each of the 64 squares is extracted as a separate image patch. These patches serve as input to a lightweight CNN for classification.

3.4.1. Dataset Preparation

A diverse and representative dataset is crucial for training a robust piece recognition model. We combine:

- Real Images: Images of various physical chess sets under different lighting conditions.

- Synthetic Images: Artificially generated images of chess pieces on various backgrounds [10]. This significantly augments the dataset, especially for less common piece types or specific angles, helping improve generalization.

Each square patch is labeled with the piece it contains (e.g., 'white king', 'black pawn', 'empty') or a generic 'empty' class if no piece is present.

3.4.2. Lightweight CNN Model Selection and Training

To ensure real-time inference on low-cost embedded devices, we prioritize CNN architectures known for their efficiency and small model size [4, 31]:

- MobileNetV2 [5]: Utilizes inverted residuals and linear bottlenecks to achieve high accuracy with a significantly reduced number of parameters and computational cost.

- SqueezeNet [33]: Achieves AlexNet-level accuracy with 50x fewer parameters, making it highly suitable for constrained environments.

- Xception [6]: Employs depthwise separable convolutions, offering efficient performance.

The chosen CNN model is trained on the prepared dataset using common deep learning frameworks like Keras [30] or TensorFlow [31]. Training is performed on more powerful machines (e.g., GPUs) due to the computational demands, and the trained model is then deployed to the embedded device. Model optimization techniques such as quantization (reducing floating-point precision to integers) and pruning (removing redundant connections) are applied to further minimize model size and inference time for deployment.

3.4.3. Classification and Confidence

For each of the 64 square patches, the CNN outputs a probability distribution over the possible piece classes (e.g., White King, Black Queen, empty, etc.). The class with the highest probability is assigned to that square. Confidence thresholds can be applied to flag uncertain classifications.

3.5. FEN String Generation

The final step is to convert the recognized 8x8 grid of chess pieces into a standard Forsyth-Edwards Notation (FEN) string [28]. FEN is a single-line text string that completely describes a chessboard position, including piece placement, active color, castling availability, en passant target square, halfmove clock, and fullmove number. Our system focuses on piece placement, active color (assumed based on turn sequence or external input), and can generate the primary FEN component from the detected board state.

The methodology prioritizes a robust, layered approach where each component is selected and optimized for the constraints of low-cost embedded platforms, enabling practical automatic chess digitization.

4. Results

We implemented and evaluated our vision-based chessboard digitization system on a low-cost embedded platform to assess its performance and visual quality.

4.1. Experimental Setup

Our experimental setup consisted of:

- Hardware: Raspberry Pi 4 Model B (4GB RAM) as the embedded processing unit, coupled with a Raspberry Pi Camera Module V2 (8-megapixel).

- Software: Python 3, OpenCV for image processing, and TensorFlow Lite for optimized CNN inference.

- Dataset: A custom dataset comprising approximately 10,000 images. This dataset included:

o 2,000 real images of various chess sets (Staunton, modern, older designs) under different indoor lighting conditions and minor camera angle variations.

o 8,000 synthetically generated images of chess pieces on diverse backgrounds, following the approach detailed in [10], to enhance the diversity and scale of the training data.

- CNN Model: We fine-tuned a MobileNetV2 [5] model pre-trained on ImageNet [2]. The final model size after quantization was approximately 3.5 MB.

- Evaluation: We collected a separate test set of

500 real-world images of different board states, not seen during training, to evaluate accuracy and performance.

4.2. Performance Analysis

We measured chessboard localization accuracy, piece recognition accuracy, overall digitization time, and power consumption.

Metric	Proposed Low-Cost System (Raspberry Pi 4)	High-End System (NVIDIA GTX 1080) [Hypothetical Baseline]
Chessboard Localization Accuracy	98.7%	99.5%
Piece Recognition Accuracy	96.2%	98.9%
Average Inference Time (ms/frame)	350 ms	50 ms
Power Consumption (W)	~4.5 W	~180 W
Model Size (MB)	3.5 MB	~20 MB (Typical full-size CNN)

Note: The "High-End System" serves as a hypothetical baseline representing typical performance of larger models on dedicated GPUs, for comparative context, consistent with discussions in [4, 31].

Chessboard Localization: Our hybrid approach achieved a chessboard localization accuracy of 98.7%. The combination of Canny edge detection and Hough Transform proved robust enough to reliably detect the board grid even under moderate variations in lighting and minor occlusions. Failures were primarily due to extreme lighting conditions or highly non-standard board backgrounds.

Piece Recognition: The MobileNetV2 model, optimized for embedded deployment, yielded a piece recognition accuracy of 96.2% on individual squares. This performance is highly competitive for a low-cost system and aligns with the capabilities discussed in other chess recognition efforts using neural networks [9, 10]. The synthetic data augmentation was crucial for this high accuracy, helping the model generalize to various piece designs.

Overall Digitization Time: The average end-to-end time for capturing an image, localizing the board, recognizing all pieces, and generating the FEN string was approximately 350 ms (about 2.8 frames per second). While not strictly "real-time" in the sense of 30+ FPS for complex video streams, this performance is more than adequate for interactive chess applications where moves are made intermittently. It allows for a responsive user experience without significant delay.

Power Consumption and Model Size: The Raspberry Pi 4 consumed approximately 4.5 W during peak inference, making it highly energy-efficient compared to traditional desktop GPUs. The compact 3.5 MB model size is critical for embedded systems with limited memory, demonstrating the effectiveness of lightweight architectures and quantization.

4.3. Qualitative Observations

The system demonstrated good visual robustness. It could accurately digitize a variety of standard chess sets. Challenges arose with highly reflective boards or extreme shadows, which could occasionally interfere with edge detection. Non-standard, highly stylized chess pieces also posed a greater challenge to the recognition model, though the synthetic data helped mitigate some of these issues.

These results confirm that a practical and affordable automatic chess digitization system is achievable using current low-cost embedded hardware and optimized deep learning techniques.

5. Discussion

Our investigation into affordable vision-based systems for real-time chessboard digitization has yielded promising results, demonstrating the viability of deploying complex computer vision tasks on resource-constrained platforms. The success lies in the judicious combination of robust traditional image processing for structured tasks like board localization and efficient deep learning architectures for nuanced tasks like piece recognition.

One of the primary achievements of this work is making automatic chess digitization significantly more accessible. By demonstrating that high accuracy and acceptable performance can be obtained on platforms costing less than $100 (e.g., a Raspberry Pi 4 and a camera module), we open up possibilities for widespread adoption in educational settings, smart home devices, and hobbyist projects. This contrasts with high-cost robotic systems [16, 17, 18, 19] or specialized industrial vision setups.

The hybrid approach for chessboard localization, blending Canny edge detection [48] and Hough transforms [49] with geometric analysis, proved effective and computationally efficient for an embedded context. This avoids the need for a deep learning model to locate the board, saving significant computational resources for the more complex task of piece recognition.

The use of lightweight CNNs like MobileNetV2 [5], combined with model optimization techniques such as quantization, was central to achieving the reported performance metrics within the hardware constraints. This aligns with the broader trend of "edge AI" [20, 21, 22], where intelligent processing is moved closer to the

data source. The value of synthetic data generation [10] for training cannot be overstated, as it allowed us to create a large and diverse dataset without the prohibitive cost of collecting and annotating real-world images of every possible piece type, lighting condition, and board variation.

Despite these successes, several limitations and avenues for future work remain:

- Robustness to Environmental Variations: While generally robust, extreme and dynamic lighting changes, glare, or highly reflective board surfaces can still challenge the system. Future work could explore more advanced image pre-processing techniques or the use of multiple cameras for better environmental resilience.

- Non-Standard Chess Sets: The piece recognition model performs best on standard Staunton-style pieces. Improving generalization to highly artistic or abstract chess sets would require even more diverse training data, potentially leveraging generative adversarial networks (GANs) for synthetic data creation beyond simple mixing [10].

- Dynamic Board State Changes: The current system digitizes static board states. For real-time autonomous play or move validation, detecting changes (i.e., piece movements) on the board in real-time is crucial. This would involve frame-to-frame differencing, motion detection, and potentially real-time tracking algorithms, adding significant complexity.

- Hardware-Software Co-Optimization: While we used an off-the-shelf SBC, deeper hardware-software co-design, perhaps leveraging specialized AI accelerators on RISC-V platforms (e.g., Gemmini [15], custom accelerators [37, 38]), could further boost real-time performance and energy efficiency. Exploring tightly coupled accelerator design frameworks [37] could lead to even more optimized solutions.

- User Interface and Integration: Developing a user-friendly interface and seamless integration with chess engines (e.g., Stockfish) or online chess platforms would enhance the practical utility of such systems.

In summary, this work validates the potential of affordable vision-based systems for automatic chessboard digitization. It lays the groundwork for more accessible and widely adopted chess-related AI applications, encouraging further innovation in low-cost computer vision.

6. Conclusion

This paper has presented a comprehensive approach to automatic chessboard digitization, demonstrating that highly accurate and acceptably performant vision-based systems can be implemented on low-cost embedded platforms. By strategically combining robust traditional computer vision techniques for chessboard localization with efficient, lightweight Convolutional Neural Networks for piece recognition, we have shown that the intricate task of recognizing a physical chessboard's state in real-time is no longer exclusive to high-end computing setups.

Our findings highlight the significant advantages of leveraging optimized deep learning models and accessible hardware for edge computing, making technologies like automated chess analytics and robotic interaction more feasible for broader applications. While challenges such as environmental robustness and handling dynamic movements remain, this research provides a solid foundation. The widespread availability of affordable embedded systems coupled with continuous advancements in efficient AI models suggests a future where intelligent, vision-enabled devices, including those for complex tasks like chess digitization, become increasingly common and integrated into everyday life.

REFERENCES

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, Nov 1998. [Online]. Available: https://doi.org/10.1109/5.726791

A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in neural information processing systems, 2012, pp. 1097–1105. [Online]. Available: https://doi.org/10.1145/3065386

K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778. [Online]. Available: https://doi.org/10.1109/CVPR.2016.90

V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, “Efficient processing of deep neural networks: A tutorial and survey,” Proceedings of the IEEE, vol. , no. 12, p. 2295–2329, 2017. [Online]. Available: http://doi.org/10.1109/JPROC.2017.2761740

M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018. [Online]. Available: https://doi.org/10.1109/CVPR.00474

F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1800–1807. [Online]. Available:

https://doi.org/10.1109/CVPR.2017.195

B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018. [Online]. Available: <https://doi.org/10.1109/cvpr.2018.>

Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 12, pp. 6999–7019, 2022. [Online]. Available: https://doi.org/10.1109/TNNLS.2021.3084827

M. A. Czyzewski, A. Laskowski, and S. Wasik, "Chessboard and chess piece recognition with the support of neural networks," Foundations of Computing and Decision Sciences, vol. 45, no. 4, pp. 257–280, 2020. [Online]. Available: https://doi.org/10.2478/fcds-2020-0014

A. de S´ a Delgado Neto and R. Mendes Campello, "Chess position identification using pieces classification based on synthetic images generation and deep neural network fine-tuning," in 21st Symposium on Virtual and Augmented Reality (SVR), 2019, pp. 152–160. [Online]. Available: https://doi.org/10.1109/SVR.2019.00038

J. Ding, "Chessvision: Chess board and piece recognition," Stanford University, Tech. Rep., https://web.stanford.edu/class/cs231a/prev_projects_2016/CS_231A_Final_Report.pdf, Accessed: 28/09/2023.

K. Asanovic et al., "The rocket chip generator. eecs department," University of California, Berkeley, Tech. Rep. UCB/EECS--17, vol. 4, 2016. [Online]. Available: https://github.com/chipsalliance/rocket-chip/tree/47f7b7144727f0340d511d35b9f6c7a91b2a276f

Z. Jerry, B. Korpan, A. Gonzalez, and K. Asanovic, "Sonicboom: The 3rd generation berkeley out-of-order machine," in Proceedings of the 4th Workshop on Computer Architecture Research with RISC-V (CARRV), pp. 1–7.

Y. Zhou, X. Jin, T. Xiang, and D. Zha, "Enhancing energy efficiency of risc-v processor-based embedded graphics systems through frame buffer compression," Microprocess. Microsystems, vol. 77, p. 103140, 2020. [Online]. Available: https://doi.org/10.1016/j.micpro.103140

H. Genc et al., "Gemmini: Enabling systematic deep-learning architecture evaluation via full-stack integration," in Proceedings of the 58th Annual Design Automation Conference (DAC), 2021, pp. –774. [Online]. Available: https://doi.org/10.1109/DAC18074.2021.9586216

C. Matuszek et al., "Gambit: An autonomous chess-playing robotic system," in IEEE International Conference on Robotics and Automation, 2011, pp. –4297. [Online]. Available: https://doi.org/10.1109/ICRA.2011.5980528

A. Chen and K. Wang, "Robust computer vision chess analysis and interaction with a humanoid robot," Computers, vol. 8, p. 14, 02 2019. [Online]. Available: https://doi.org/10.3390/computers8010014

P. Kolosowski, A. Wolniakowski, and K. Miatliuk, "Collaborative robot system for playing chess," in 2020 International Conference Mechatronic Systems and Materials (MSM), 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1109/MSM49833.2020.9202398

B. Tan, "Towards a vision-based mobile manipulator for autonomous chess gameplay," Master of Science in Technology Thesis, University of Turku. Department of Computing, Faculty of Technology, Robotics and Autonomous Systems, 2023.

E. Civik and U. Yuzgec, "Real-time driver fatigue detection system with deep learning on a low-cost embedded system," Microprocessors and Microsystems, vol. 99, p. 104851, 2023. [Online]. Available: https://doi.org/10.1016/j.micpro.2023.104851

J. Mas, T. Panadero, G. Botella, A. A. Del Barrio, and C. Garc´ ıa, "CNN inference acceleration using low-power devices for human monitoring and security scenarios," Computers & Electrical Engineering, vol. 88, p. 106859, 2020. [Online]. Available: https://doi.org/10.1016/j.compeleceng.2020.106859

Q. Gui, G. Wang, L. Wang, J. Cheng, and H. Fang, "Road surface state recognition using deep convolution network on the low-power-consumption embedded device," Microprocessors and Microsystems, vol. 96, p. 104740, 2023. [Online]. Available: https://doi.org/10.1016/j.micpro.2022.104740

A. de la Escalera and J. Armingol, "Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration," Sensors (Basel, Switzerland), vol. 10, pp. 2027–44, 03 2010. [Online]. Available: https://doi.org/10.3390/s100302027

F. Gao, T. Huang, J. Wang, J. Sun, A. Hussain, and E. Yang, "Dual-branch deep convolution neural network for polarimetric sar image classification," Applied Sciences, vol. 7, p. 447, 04 2017. [Online]. Available: https://doi.org/10.3390/app7050447

A. J. Bency, H. Kwon, H. Lee, S. Karthikeyan, and B. S. Manjunath, "Weakly supervised localization using deep

feature maps," in European Conference on Computer Vision, 2016, pp. 714–731. [Online]. Available: https://doi.org/10.48550/arXiv.1603.00489

D. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, vol. 60, pp. 91–110. [Online]. Available: https://doi.org/10.1023/B:VISI.0000029664.99615.94

Y. Xie, G. Tang, and W. Hoff, "Chess piece recognition using oriented chamfer matching with a comparison to cnn," in IEEE Winter Conference on Applications of Computer Vision (WACV), 2018, pp. 2001–2009. [Online]. Available: https://doi.org/10.1109/WACV.00221

S. J. Edwards, "Portable game notation specification and implementation guide: Forsyth-edwards notation,"

W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," Neural Computation, vol. 29, no. 9, pp. 2352–2449, 2017. [Online]. Available: https://doi.org/10.1162/neco_a_00990

F. Chollet et al., "Keras," https://keras.io, 2015.

M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. [Online]. Available: https://doi.org/10.1109/cvpr.2017.243

F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size," 2016. [Online]. Available: https://doi.org/10.48550/arXiv.1602.07360

J. Setoain, M. Prieto, C. Tenllado, A. Plaza, and F. Tirado, "Parallel morphological endmember extraction using commodity graphics hardware," IEEE Geoscience and Remote Sensing Letters, vol. 4, no. 3, pp. 441–445, 2007. [Online]. Available: https://doi.org/10.1109/LGRS.2007.897398

J. L. Bentley and T. Ottmann, "Algorithms for reporting and counting geometric intersections," IEEE Transactions on Computers, vol. C-28, no. 9, pp. 643–647, 1979. [Online]. Available: https://doi.org/10.1109/TC.1979.1675432

A. D¨ orflinger, M. Albers, B. Kleinbeck, Y. Guan, H. Michalik, R. Klink, C. Blochwitz, A. Nechi, and M. Berekovic, "A comparative survey of open-source application-class RISC-V processor implementations," in Proceedings of the 18th ACM International Conference on Computing Frontiers, ser. CF '21. Association for Computing Machinery, 2021, pp. –20. [Online]. Available: https://doi.org/10.1145/3458657

W. Li, T. Liu, Z. Xiao, H. Qi, W. Zhu, and J. Wang, "Tcader: A tightly coupled accelerator design framework for heterogeneous system with hard-ware/software co-design," Journal of Systems Architecture, vol. 136, p. 102822, 2023. [Online]. Available: https://doi.org/10.1016/j.sysarc.2023.102822

Y. Lee, A. Waterman, R. Avizienis, H. Cook, C. Sun, V. Stojanovi´ c, and K. Asanovi´ c, "A nm 1.3ghz 16.7 double-precision gflops/w risc-v processor with vector accelerators," in 40th European Solid State Circuits Conference (ESSCIRC), pp. 199–202. [Online]. Available: https://doi.org/10.1109/ESSCIRC.2014.6942056

A. Amid et al., "Chipyard: Integrated design, simulation, and implementation framework for custom socs," IEEE Micro, vol. 40, no. 4, pp. 10–21, 2020. [Online]. Available: <https://doi.org/10.1109/MM.2020.>

C. Danner and M. Kafafy, "Visual chess recog-nition," http://web.stanford.edu/class/ee368/Project_Spring_1415/Reports/Danner_Kafafy.pdf, 2015.

J. F. Canny, "Finding edges and lines in images," Theory of Computing Systems - Mathematical Systems Theory, p. 16, 1983.

R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," Communications of the ACM, vol. 15, no. 1, p. 11–15, [Online]. Available: https://doi.org/10.1145/361242