Development of AI-based control system for 6-axis robotic arm

SUP MICROTECH ENSMM
ECOLE NATIONALE SUPERIEURE DE MECANIQUE ET DES MICROTECHNIQUES

Björn Andrews
bjoern.andrews@eu4m.eu

Supervisors:
Juan Carlos Alvarez Alvarez, UNIOVI, juan@uniovi.es
Nadine Piat, ENSMM, nadine.piat@ens2m.fr

eu4m
Erasmus Mundus Master in Mechatronic Engineering

Abstract

Contemporary industrial robotic systems are evolving beyond routine tasks to require agile responses to diverse object shapes. Leveraging the surge in computational power and affordable sensor technology, computer vision-driven systems offer a solution. This project presents a pick and place system using computer vision, featuring a UR3e robot with an RGI14 gripper and an Intel RealSense L515 RGB-D camera. A simulation model is first created to validate functionality, developed using RoboDK and MATLAB. The RealSense MATLAB wrapper and RoboDK API enable seamless communication between the components. A tiny YOLOv4 network trained with synthetic data detects objects, while a PCA-based algorithm by MathWorks estimates their poses. Demonstrated in simulation and real setup, the system identifies plastic bricks using color image and depth data, allowing accurate sorting. While the simulation shows robust functionality, occasional real-world inaccuracies occur due to limited synthetic data's impact on training the object detection network for real objects.

Résumé

Les systèmes robotiques industriels contemporains évoluent au-delà des tâches routinières et exigent des réponses agiles à des objets de formes diverses. En tirant parti de l'augmentation de la puissance de calcul et de la technologie abordable des capteurs, les systèmes basés sur la vision par ordinateur offrent une solution. Ce projet présente un système de prise et de dépose utilisant la vision par ordinateur, avec un robot UR3e équipé d'une pince RGI14 et d'une caméra RGB-D Intel RealSense L515. Un modèle de simulation est d'abord créé pour valider la fonctionnalité, développé à l'aide de RoboDK et de MATLAB. Le wrapper MATLAB de RealSense et l'API RoboDK permettent une communication transparente entre les composants. Un petit réseau YOLOv4 entraîné avec des données synthétiques détecte les objets, tandis qu'un algorithme basé sur l'ACP de MathWorks estime leurs positions. Démontré en simulation et en configuration réelle, le système identifie les briques en plastique à l'aide d'images en couleur et de données de profondeur, ce qui permet un tri précis.

Keywords: Universal Robots, MATLAB, RoboDK, Computer Vision

1. Motivation and Objectives + State of the art

Upon arrival at the laboratory, the Universal Robots UR3e model was initially utilized without a gripper, employing markers for movement and control (Figure 1). The project's motivation was to enhance the system by integrating a gripper and camera, creating an AI-driven handling system capable of flexible object detection and manipulation through computer vision. The overarching goal included establishing a digital twin for testing computer vision and robot movements prior to actual implementation. Additionally, the preference within the laboratory was to develop the control program in MATLAB, enabling easy expansion and simulation through tools like the Simscape Toolbox.

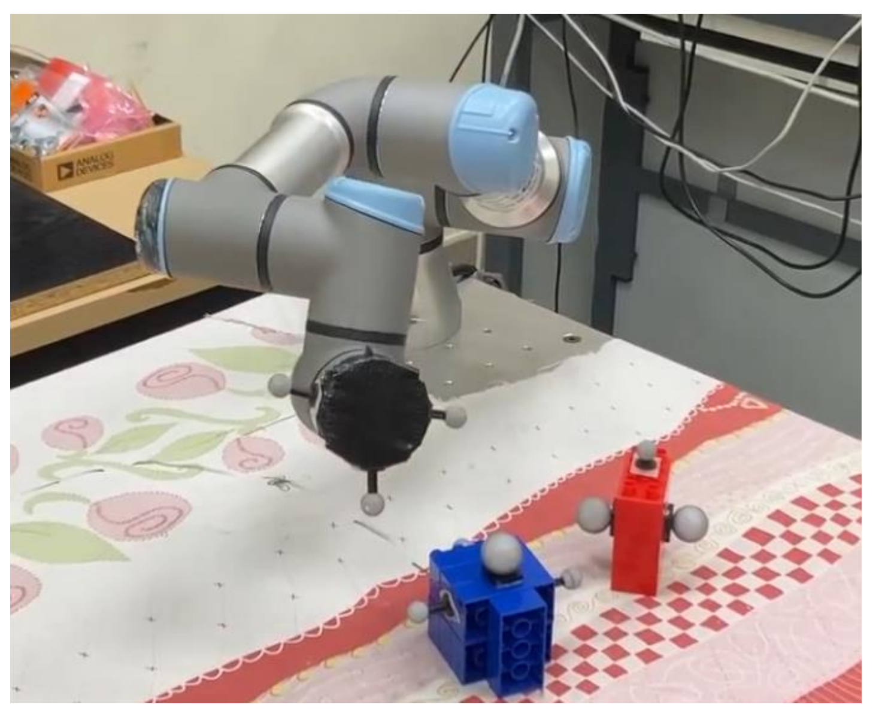
Figure 1: initial state in laboratory

MathWorks has developed various applications to highlight MATLAB and Simulink capabilities, one being an Intelligent Bin Picking system that merges perception and motion planning (Figure 2). This system employs MATLAB's Computer Vision Toolbox to employ deep learning for object position and orientation detection within a bin. Subsequently, a motion planning algorithm uses calculated object poses to generate trajectory paths for a Universal Robots UR5e robot's pick-and-place actions. Communication with the UR5e robot is established through the Robotics System Toolbox Support Package for Universal Robots UR Series Manipulators. Object detection relies on a pre-trained YOLOv4 network for efficient processing. Pose calculation for objects utilizes a PCA-based algorithm.

Figure 2: Bin Picking application by MathWorks

2. Development of AI-based handling system

The project's system architecture comprises key elements (Figure 3): the UR3e robot for dynamic interactions, an Intel RealSense L515 camera for image and depth data, a RGI14 gripper for object manipulation, suitable objects like LEGO bricks, and a computer for computation. USB 3.2 and RS232 enable camera and robot connectivity. The MATLAB main program oversees operations, employing the RealSense MATLAB Wrapper for camera communication. It also collaborates with RoboDK, which interfaces with the UR3e robot.

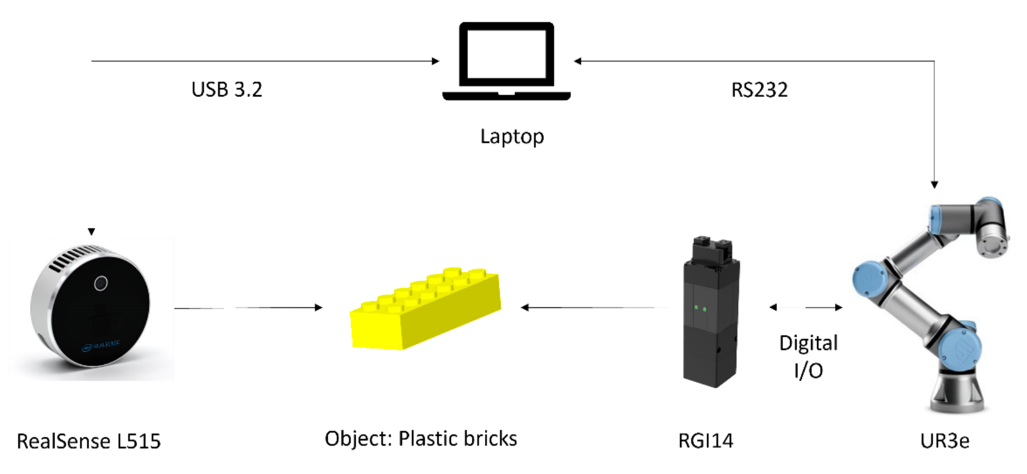

Figure 3: system architecture

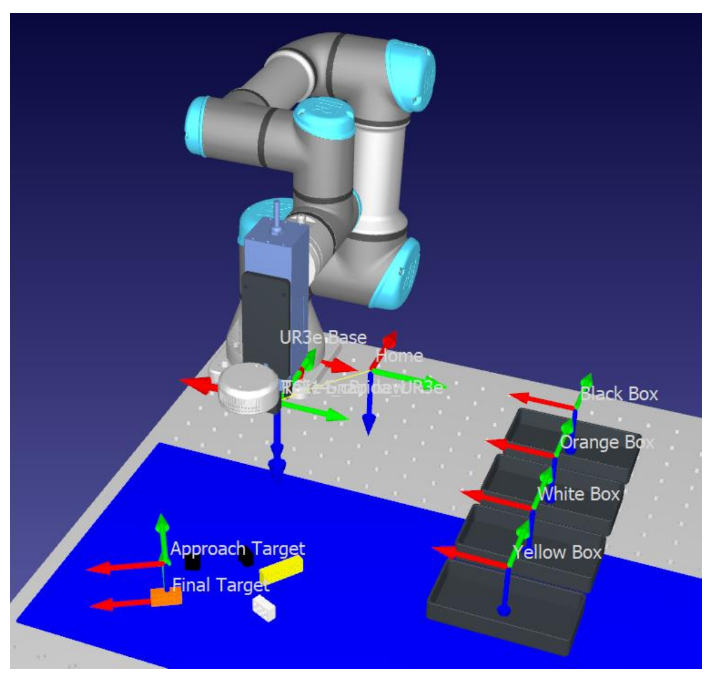

Figure 4: robotics simulation model

The RoboDK software facilitates the creation of a simulation model for the pick and place application (Figure 4), allowing testing without physical robot, camera, or gripper. The RoboDK station includes 3D models of the UR3e robot, a table, gripper, camera, LEGO bricks, and self-designed flanges and sorting boxes. The simulation enables realistic robot movements and collision detection, while a simulated color and depth camera mimics RealSense L515 functionality. Movement instructions are translatable to the UR3e robot controller through a post processor, and can also be executed on the actual robot using TCP/IP protocol.

The simulation model is also used for generating synthetic image data of the bricks for training a tiny YOLOv4 object detection network.

The object detection network, already trained, is employed to identify differently colored bricks placed randomly in images. Predicted bounding boxes containing brick classifications and their image coordinates are the network's output. Pose estimation is conducted using depth data as a point cloud in MATLAB. A function segregates planes for bricks and the table using the depth data. Remaining brick point clouds within bounding box regions are individually analyzed via PCA (principal component analysis) to extract the orientation.

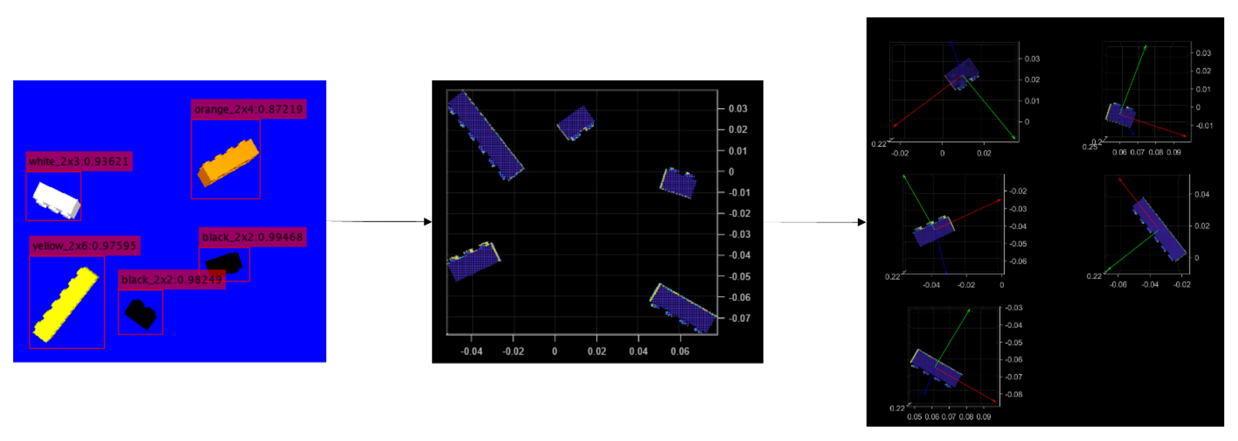
Figure 5: object detection and pose estimation

3. Results of simulation model and real setup

The real hardware setup mirrors the simulation model precisely, translating simulated instructions to the real setup (Figure 6). Although dimensions largely matched, a slight 3mm adjustment was made to the camera position for enhanced accuracy.

Each brick iteration involves the robot moving its TCP to an approach position, then to the target position for collision avoidance. The gripper closes to grasp the brick, then returns to approach and initial positions to avoid collisions. Bricks are sorted based on their color label.

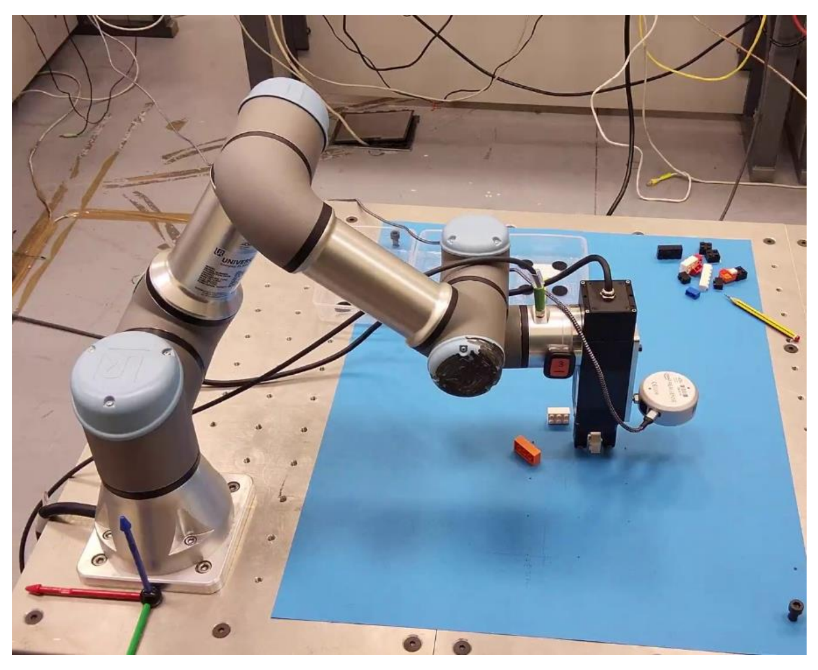
Figure 6: real hardware setup

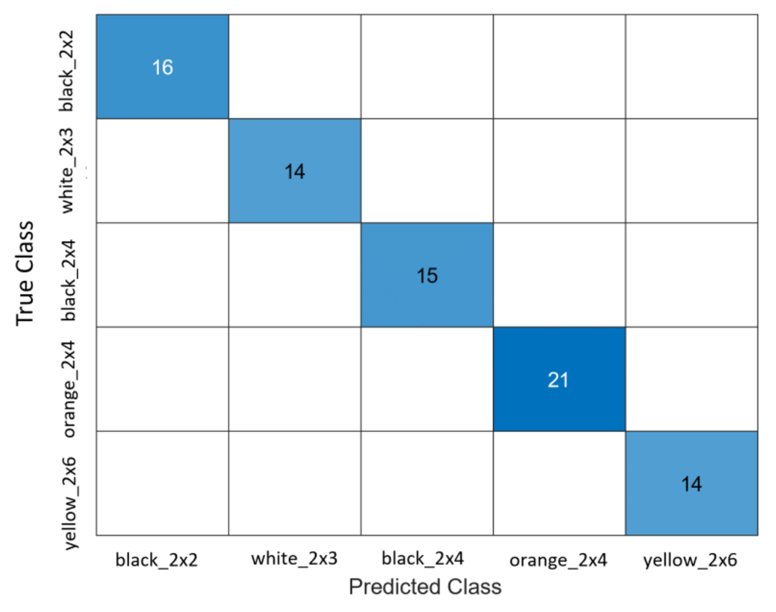

Figure 7: confusion matrix of simulation model

The simulation model validates its functionality, accurately detecting randomly generated bricks and demonstrating correct sorting. A confusion matrix (Figure 7) reveals precise object detection results for simulated images, while accurate pose estimation prevents collisions during gripping. The trained object detector reliably classifies bricks for correct color-based sorting. Transitioning to the real setup, the model's effectiveness persists, with most bricks being detected using RealSense L515's image data. Proper position estimation ensures successful gripping of detected bricks.

While the simulation yielded perfect results, challenges emerged with object detection in the real setup. The YOLOv4 network, trained solely on synthetic data, exhibited less than 100% accuracy, occasionally failing to detect bricks like the white one shown in Figure 8. Detection errors weren't confined to a specific color but occurred randomly across all colors. Even when detected, misclassifications arose, particularly with orange bricks wrongly classified as yellow. A viable solution might involve retraining the object detection network using labeled real image data to improve stability, considering diverse lighting conditions and background textures.

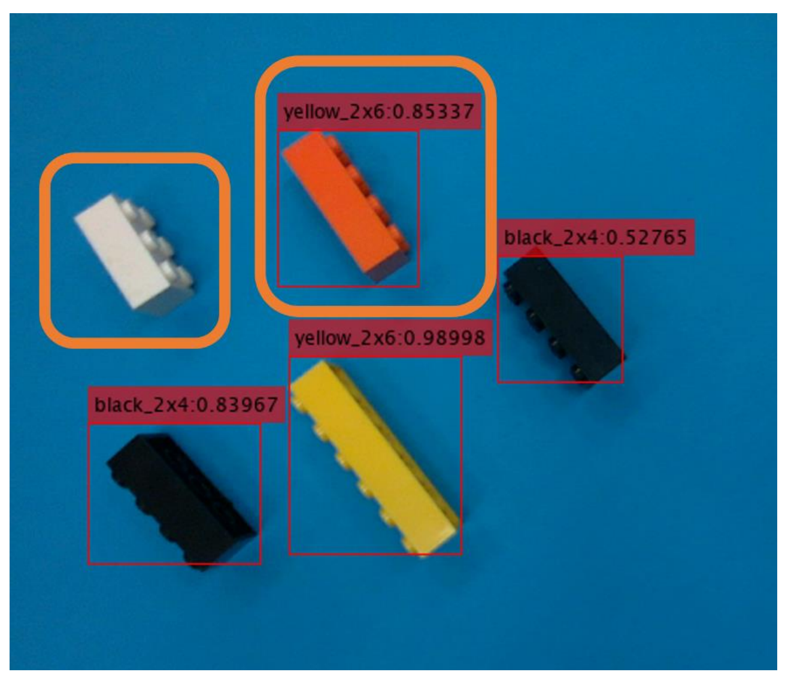

Figure 8: errors in real object detection

4. Conclusions

1. Realization of pick and place application with robotic arm, gripper and AI-based object detection.
2. Creation of robotic simulation model with robust system functionality. Functionality of pick and place system with Real-world setup validation.
3. Improvement for object detection accuracy and classification reliability in real setup necessary.
4. Potential for future projects in a specific field; suggestions: diversifying training data, extending to 3D problems, using integrated camera for collision detection, enhancing gripper's design

References

- MathWorks, "Intelligent Bin Picking in MATLAB® for Universal Robots," 2023. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/125240-intelligent-bin-picking-in-matlab-for-universal-robots?s_tid=answers_rc2-2_p5_BOTH.
- P. Corke, "Robotics, Vision, Control," Springer Cham, 2017.
- RoboDK, "RoboDK Matlab API," 2023. [Online]. Available: https://robodk.com/Matlab-API.
- Intel RealSense, "Intel Realsense Matlab Wrapper," 2023. [Online]. Available: https://dev.intelrealsense.com/docs/matlab-wrapper.

Acknowledgements

I would like to thank my supervisors of this project. Especially Juan Carlos Alvarez Alvarez for guiding me with all the ideas, for giving me the opportunity to use all the equipment in his laboratory, and for providing me with additional components needed.