Prototyping Educational and Scientific Devices with a Custom Python Library for Lego Robot Inventor 5in1 Mindstorms Kit: A Leap Motion Integration Case Study

Jakub Malinowski[1], Piotr Artiemjew[2][a] and Wojciech Cybowski[1]

[1]*Scientific Circle of Robotics, Univeristy of Warmia and Mazury in Olsztyn, Poland*

[2]*University of Warmia and Mazury in Olsztyn, Poland*

Keywords: Educational Technology, Prototyping Tools, Python Programming, Lego Mindstorms, Leap Motion Controller, Interactive Learning Environments.

Abstract: This paper introduces a custom Python library specifically developed to enable the rapid prototyping of devices for educational and scientific purposes using the Lego Mindstorms Robot Inventor set. The integration of Leap Motion as a case study exemplifies the library's extensive capabilities in facilitating intuitive and interactive control mechanisms for mobile robots. Through the Leap Motion controller, this research explores the innovative application of real-time hand gesture recognition, allowing users to command the robot with natural gestures for movement, direction, and speed adjustments. The decision to use Leap Motion as a case study serves to highlight the library's adaptability and effectiveness in processing and interpreting gesture data, thereby offering a glimpse into the broader potential of the library for various applications. This approach demonstrates how educators and researchers can utilize the library to create engaging learning experiences and conduct exploratory projects in robotics and beyond. By detailing the process of integrating Leap Motion with the Robot Inventor set through our Python library, the paper underscores the potential of such tools in enhancing interactive learning environments and advancing the field of educational technology. The open-source nature of the library, coupled with its modular design, ensures that it can be easily extended and adapted to fit a wide range of educational and scientific prototyping needs. This paper thus presents a valuable resource for those looking to explore the frontiers of interactive technology in education and research.

1 INTRODUCTION

The use of prototyping in education and research is a very valuable practice. In general, it provides an opportunity to teach in an interesting way, by observing the effects in the real world. And to create low-cost initial versions of devices to verify research hypotheses. The current work is located in the area of application of prototyping in learning to develop AI techniques and creating hybrid IoT and robotic systems for research scenarios. In our laboratory (UWM, 2024), we have many years of experience in the use of hardware in teaching intelligent robotics and artificial intelligence. Among others, we specialise in the development of gesture control systems, having experience in controlling Roomba, Nao, Spot, lego robots - see for example the work (Żmudziński et al., 2016). We develop systems to train robotics and AI algorithms using augmented reality markers - see for example (Szpakowska et al., 2023). We generally develop technology for algorithm learning and research.

In this paper, we focus on connecting the Robot Inventor kit with the Leap Motion controller. We use a Python library we developed in our robotics lab (Cybowski, 2023) to help the robot work with a dedicated Leap Motion control software suite. This work not only highlights how gesture control can be applied in robotics but also shows our lab's dedication to pushing forward the development of educational and scientific devices with easy-to-use and accessible technology

The next sections of this paper have the following content. In sections 1.1 and 1.2 we have a discussion of the potential applications of our system in education and research. In section 2, we discuss the individual components of the project, the implementation part and demonstration of the project. Finally, we summarise the work and discuss future works.

[a] https://orcid.org/0000-0001-5508-9856

Malinowski, J., Artiemjew, P. and Cybowski, W.
Prototyping Educational and Scientific Devices with a Custom Python Library for Lego Robot Inventor 5in1 Mindstorms Kit: A Leap Motion Integration Case Study.
DOI: 10.5220/0012712600003693
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Conference on Computer Supported Education (CSEDU 2024) - Volume 1*, pages 542-549
ISBN: 978-989-758-697-2; ISSN: 2184-5026
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

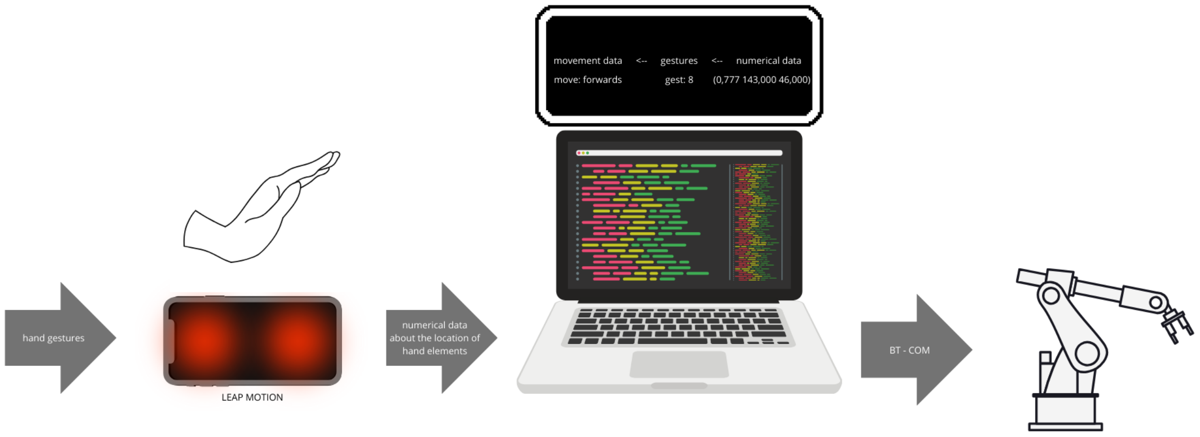
movement data <-- gestures <-- numerical data
move: forwards gest: 8 (0,777 143,000 46,000)
hand gestures
LEAP MOTION
numerical data about the location of hand elements
BT - COM

Figure 1: Graphical presentation of the system - use of Leap motion as a case study.

Let's start by discussing the potential applications of our lego mindstorms robot inventor 5in1 control python library in educational contexts.

1.1 Educational Applications

Robotics Education. The developed Python library and Leap Motion integration offer significant advantages in robotics education. By providing students with a hands-on experience in controlling a robot through intuitive hand gestures, this technology enhances the learning process (Anderson and Dill, 2000). It promotes a deeper understanding of robotics concepts, and sensor integration. Students can experiment with different gestures and observe their immediate impact on robot behavior, cultivating a practical understanding of robotics principles.

Engaging STEM Learning. The use of gesture-based control makes learning about science, technology, engineering, and mathematics (STEM) more engaging and interactive. It attracts the interest of students at various educational levels, from elementary schools to universities. The tool can be integrated into STEM curriculum modules, enabling educators to teach abstract concepts in a real and exciting manner (Folk, 1981).

Prototyping and Design Projects. Students can use the Python library to design and build their robotic prototypes, experimenting with different control strategies. This hands-on approach encourages creativity and problem-solving skills (Resnick et al., 2009). Additionally, it allows students to gain practical experience in the iterative design process, from concept development to prototype testing and refinement.

Interdisciplinary Learning. The flexibility of the Python library and its compatibility with other devices and sensors create opportunities for interdisciplinary learning. Students from diverse backgrounds, including computer science, engineering, and design, can collaborate on projects that incorporate both hardware and software elements (Cagiltay et al., 2014). This promotes a comprehensive understanding of technology integration.

Accessible Learning. The open-source nature of the library ensures accessibility, making it an inclusive tool for diverse learners. It can be easily shared among educational institutions, ensuring that students from various backgrounds have equal opportunities to explore robotics and human-computer interaction (Miglino et al., 1999).

Research Collaboration. Educators and researchers can collaborate on innovative teaching methods and curricula development using this technology. By sharing experiences and best practices, they can continually improve robotics education and contribute to the growth of the field.

In summary, the developed Python library and Leap Motion controller integration offer valuable resources for enhancing robotics education. They provide a platform for engaging, interactive, and interdisciplinary learning experiences, empowering students to explore the exciting world of robotics while encouraging creativity, problem-solving skills, and collaboration. Moreover, the open-source nature of the tool promotes accessibility and collaboration among educators and researchers, further advancing the field of robotics education.

Let's move on to discuss an overview of the research and teaching applications of our project results.

1.2 Research Applications

Rapid Prototyping and Experimentation. The Python library and Leap Motion controller integration provide researchers with a resourceful platform for rapid prototyping and experimentation in the field of robotics and human-machine interaction. Researchers can quickly develop and test novel control algorithms, user interfaces, and interaction paradigms, reducing development time and costs (Lawitzky et al., 2013).

Human-Robot Interaction Studies. The gesture-based control system enables researchers to conduct in-depth studies on human-robot interaction (HRI). By using hand gestures to control robots, researchers can investigate how users perceive and interact with robotic systems. This technology can be applied in various HRI contexts, such as healthcare, manufacturing, and service robotics (Neto et al., 2019).

Usability Testing. Researchers can use the tool to conduct usability testing of robotic systems and interfaces. By analyzing user interactions and feedback, they can fine-tune the design of robotic applications to improve user experience and efficiency. This is particularly valuable in the development of robotic solutions for specific industries and user groups (Chacón et al., 2021).

Sensor Integration and Data Collection. The Python library's compatibility with various sensors and devices allows researchers to integrate additional hardware components seamlessly. This flexibility enables the collection of rich sensor data, facilitating research in areas such as environmental monitoring, object recognition, and spatial awareness (Zhang and Doyle, 2023).

Collaborative Research. The open-source nature of the library promotes collaboration among researchers and institutions. Researchers can share their code and experiences, promoting a community of experts in robotics and human-machine interaction. Collaborative efforts can lead to the development of standardized interfaces and best practices in the field (Matthews and Greenspan, 2020).

Cross-Disciplinary Research. The tool's adaptability extends its utility to cross-disciplinary research projects. Researchers from diverse fields, including robotics, computer science, psychology, and engineering, can collaborate on projects that use gesture-based control and robotics technology to address complex research questions (Rex Hartson, 1998).

Validation and Benchmarking. Researchers can use the library to validate and benchmark their control algorithms and robotic systems. By comparing the performance of their solutions with the gesture-based control system, they can assess the effectiveness of their approaches and contribute to the advancement of the field (Araujo et al., 2023).

In conclusion, the Python library and Leap Motion controller integration offer researchers a powerful tool for advancing their work in robotics and human-machine interaction. Whether it's for rapid prototyping, in-depth HRI studies, usability testing, or cross-disciplinary research, this technology provides a flexible and accessible platform. The open-source nature of the tool encourages collaboration and knowledge-sharing among researchers, driving innovation in the field and contributing to the development of robust and user-friendly robotic systems.

Now lets discuss in detail how we created our smart gesture control system, which is a demonstration of the use of our library.

2 PROJECT COMPONENTS AND IMPLEMENTATION

Lets start by discussing the hardware compotents used in the work.

2.1 Description of the Leap Motion Controller

Leap Motion is a motion sensing and hand tracking technology designed to provide precise and intuitive control of digital devices through natural hand and finger movements (Ultraleap, 2023). It utilizes advanced cameras and infrared sensors to track hand and finger movements with remarkable accuracy, offering a wide field of view of approximately 135 degrees. Operating at up to 150 frames per second, it captures hand gestures and movements in real time, making it compatible with both Windows and macOS operating systems. Leap Motion is equipped with a software development kit (SDK), enabling developers to integrate hand tracking into various applications, including virtual reality, augmented reality, and interactive design. It finds applications in fields such as virtual reality gaming, medical simulations, architectural design, and more, enhancing user interaction and immersion in digital environments.

2.2 Description of Robot Inventor Kit

The LEGO Robot Inventor 5in1 Mindstorms kit is a flexible and educational robotics set that allows users to build and program five different robots (LEGO Group, 2023). It offers a wide range of components, including motors, sensors, and a programmable hub, enabling users to create robots capable of various tasks. With the help of the intuitive LEGO Mindstorms software, users can code their robots to perform actions, follow commands, and even respond to sensors. This kit is designed to teach programming, engineering, and problem-solving skills in a fun and hands-on way, making it suitable for both beginners and more experienced robotics enthusiasts.

2.3 Competitive Solutions

There are many robotics platforms that can provide an excellent environment to practice artificial intelligence and robotics algorithms. Among these we have, for example, the Aduino and the Raspberry Pi. The advantage of Lego robots over other solutions is that they have a closed, safe design, are very fault-tolerant and have good quality sensors and servomotors in relation to their price. In addition, by controlling Lego robots in a semi autonomous way, serious robotics algorithms can be tested at university level. For these two reasons alone, we ourselves use these robots in the process of educating students and decided to use one of them in our work.

Next, let's demonstrate how to use our library to control robots with lego 5in1 robot inventor kits and gesture recognition system.

2.4 Robot Control Using the Python Library le_mind_controller

The library, named le_mind_controller and available at the following address as open source software: github.com/wcyb/le_mind_controller, enables communication and control of the hub, a key component of the Lego Mindstorms set numbered 51515. In addition, it should also work with the hub from the Lego Spike Prime set numbered 45678, due to the fact that the hubs in the two sets differ only in external appearance, but tests have not been conducted on the hub from the 45678 set. Connection to the hub can be made via a USB cable as well as via Bluetooth. The type of connection does not affect the operation and use of the library. The library is divided into four modules:

1. Helpers.py - contains helper functions, responsible for listing the serial ports available on the system and for establishing a connection through the selected port. The open source pySerial library, available at the following address: github.com/pyserial/pyserial, is responsible for the technical, operating system-dependent aspects of handling serial ports.
2. MindComm.py - is responsible for formatting and sending control commands to the hub. It also receives responses and data sent by the hub, and then directs them to a parsing function in another module. When sending commands, it is important to remember that each must contain an individual identifier. It is randomly generated, has a length of four characters and consists of uppercase and lowercase letters, numbers and ”-_” characters. When the hub executes a command, it sends back a message with the same identifier as the command sent. This makes it easy to control the status of command execution.
3. MindData.py - processes the data received from the hub, as well as contains the definition of constant values used by the modules connected to the hub and the hub itself, such as the color seen by the sensor or the type of module connected. The functions contained in this module allow to easily obtain the information of interest from the hub itself as well as the modules connected to it.
4. SerComm.py - it is used to handle events related to the connection with the hub. Here one can find, among others, functions called in case of connection loss or receiving a new line of data from the hub.

2.5 Gesture Recognition Based on Leap Motion

The code snippet in Figure 2 is an implementation for handling data from the Leap Motion device in Python. In brief, this code analyzes data related to hand and wrist movements provided by the device and identifies certain gestures based on that data.

The rotation angles (roll) and directions (yaw, pitch) are transformed from radians to degrees. Then, the code checks the type of hand (left or right) and performs different conditional checks based on the range of the hand's rotation angle.

The main gestures identified by the code include wrist movements, hand tilt angles, and the detection of a fist by analyzing the finger angle. The results are then sent to the console and logged to a file named ”log.txt” in the form of gesture labels and angle values.

```
rolldeg = normal.roll * Leap.RAD_TO_DEG
arm = hand.arm
content1 = ""
content2 = ""
if handtype == "Right hand":
  if -125 <= rolldeg <= -45:
    content = "gestp:2"
    armdeg = arm.direction * Leap.RAD_TO_DEG
    handdir = direction.yaw * Leap.RAD_TO_DEG
     ↪  - armdeg[0]
  if -15 < handdir < 15:
    content1 = "gest:3"
    content2 = handdir
  #gest left, driving left
  elif handdir < -15:
    content1 = "gest:4"
    content2 = handdir
  #gest right, driving right
  elif handdir > 15:
    content1 = "gest:5"
    content2 = handdir
  elif -45 <= rolldeg <= 45:
    armdeg = arm.direction * Leap.RAD_TO_DEG
    handdeg = direction.pitch *
     ↪  Leap.RAD_TO_DEG - armdeg[1]
    content = "gestp:1"
    finger = hand.fingers[2]
    bone = finger.bone(3)
    punch = - bone.direction[1] *
     ↪  Leap.RAD_TO_DEG - direction.pitch *
     ↪  Leap.RAD_TO_DEG
  if 5 < handdeg < 35:
    content1 = "gest:6"
    content2 = handdeg
  #gest down, driving backward
  elif handdeg < 5:
    content1 = "gest:7"
    content2 = handdeg
  #gest up, driving forward
  elif handdeg > 35:
    content1 = "gest:8"
    content2 = handdeg
  #reset or exit gest
  if punch < -40:
    content1 = "gest:0"
    content2 = punch
  else:
    print "gest not found"
    content = False
  if content:
    print content, "\n", content1, "\n",
     ↪  "degvalue:", content2
    with open("log.txt", "a") as file:
    file.write(content + "\n")
% \end{lstlisting}
```

Figure 2: Gesture recognition based on data from leap motion.

The gestures used in the work can be seen in Figs. 3 through 10.

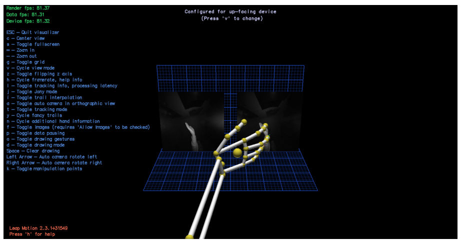
Figure 3: Gest right, driving right.

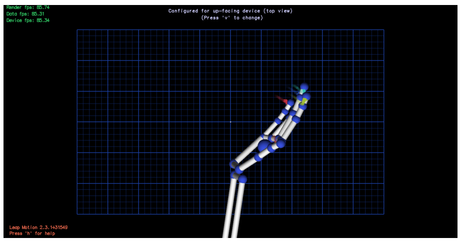
Figure 4: Gest right, driving right.

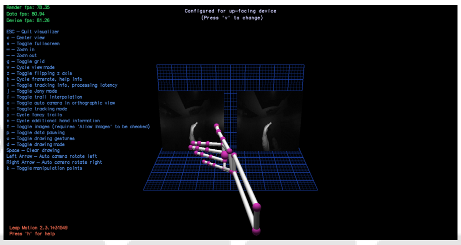
Figure 5: Gest left, driving left.

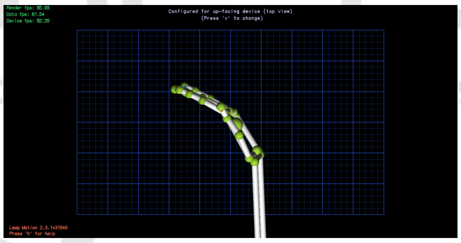
Figure 6: Gest left, driving left.

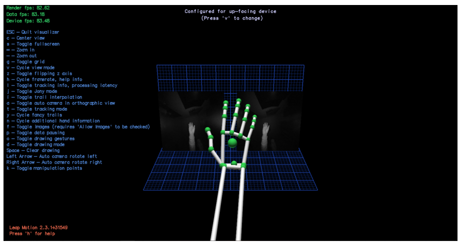
Figure 7: Gest up, driving forward.

2.6 Controlling the Robot with Gestures

The provided code snippet is responsible for receiving data from a process, analyzing the received messages regarding types of gestures or angle values, and taking appropriate actions based on the received commands. This code is dedicated to interacting with a device,

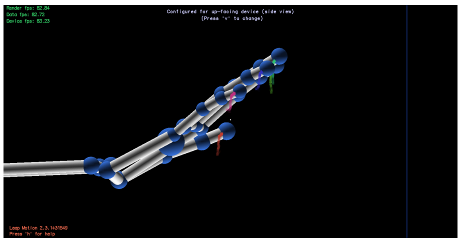

Figure 8: Gest up, driving forward.

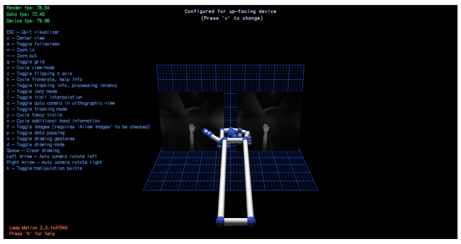
Render fps: 78.54
Data fps: 72.45
Device fps: 79.98
Configured for up-facing device
(Press 'v' to change)
ESC – Quit visualizer
c – Center view
s – Toggle fullscreen
= – Zoom in
- – Zoom out
g – Toggle grid
v – Cycle view mode
z – Toggle flipping z axis
h – Cycle framerate, help info
l – Toggle tracking info, processing latency
j – Toggle Jony mode
i – Toggle trail interpolation
a – Toggle auto camera in orthographic view
t – Toggle tracking mode
y – Cycle fancy trails
n – Cycle additional hand information
f – Toggle images (requires 'Allow Images' to be checked)
p – Toggle data pausing
o – Toggle drawing gestures
d – Toggle drawing mode
Space – Clear drawing
Left Arrow – Auto camera rotate left
Right Arrow – Auto camera rotate right
k – Toggle manipulation points
Leap Motion 2.3.1+31549
Press 'h' for help

Figure 9: Gest down, driving backward.

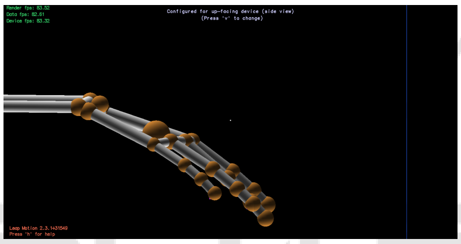

Figure 10: Gest down, driving backward.

likely related to movement or steering a vehicle.

In the code, each received line is processed to extract gesture type or angle value. Depending on the identified gesture, corresponding actions are taken, such as controlling motors to move in different directions.

The code involves checking and comparing the received commands, performing specific actions for each command, and controlling motors accordingly. The overall functionality is geared towards the control and movement of a device based on gestures and angles received from an external source.

2.7 Demonstration and Codes

For the project, we used a robot control library developed in our scientific team (Cybowski, 2023), based on the intelligent lego cube from the Robot Inventor kit (LEGO Group, 2023). And we used the SDK of the LeapMotion hand tracking device (Ultraleap, 2023). A demonstration of how to use the system is available at the youtube link (Malinowski and Artiemjew, 2023). Screen shots of the gestures used can be seen in figures from 3 to 10. The codes applied to the controls in Figures 2, 11 and 12. An example robot

```
process = subprocess.Popen(
[python2_path, "./Development/Sample.py"],
stdin=subprocess.PIPE,
stdout=subprocess.PIPE,
stderr=subprocess.PIPE,
universal_newlines=True)
[...]
for line in process.stdout:
 if line != "\n":
  # Divide the line by gesture type or
  ↪  angle value
  line_gest = line.split(":", 1)
  if line_gest[0] == "gest":
   command = line_gest[1]
   if command:
    print("Command:", command, end="")
    # Check the order received and take
    ↪  appropriate action
    # gest up, driving forward
    if command == '8 \n':
     #Taking action on change of command
     if last_command and last_command != '8
     ↪  \n':
        print('up')
        mc.cmd_stop_program_execution()
        mc.cmd_motor_turn_on(HubPortName.A,
        ↪  -50)
        mc.cmd_motor_turn_on(HubPortName.E,
        ↪  50)
        status = True
    # gest down, driving backward
    elif command == '7 \n':
     if last_command and last_command != '7
     ↪  \n':
        print('down')
        mc.cmd_stop_program_execution()
        mc.cmd_motor_turn_on(HubPortName.A,
        ↪  50)
        mc.cmd_motor_turn_on(HubPortName.E,
        ↪  -50)
        status = True
    # gest left, driving left
    elif command == '4 \n':
     if last_command and last_command != '4
     ↪  \n':
        print('left')
        mc.cmd_stop_program_execution()
        mc.cmd_motor_turn_on(HubPortName.A,
        ↪  50)
        mc.cmd_motor_turn_on(HubPortName.E,
        ↪  50)
        status = True
```

Figure 11: Robot control using leap motion data - code snipet part1.

from the Robot Inventor kit in fig. 13.

2.8 Possibilities for Other Experiments

To expand the awareness of how our system can be used we will have a discussion in this section about

```
    # gest right, driving right
    elif command == '5 \n':
     if last_command and last_command != '5
      ↪  \n':
        print('right')
        mc.cmd_stop_program_execution()
        mc.cmd_motor_turn_on(HubPortName.A,
         ↪   -50)
        mc.cmd_motor_turn_on(HubPortName.E,
         ↪   -50)
        status = True
    else:
     if status:
        # Stopping the execution of a
         ↪   command
        mc.cmd_stop_program_execution()
        status = False
    last_command = command

if line_gest[0] == "degvalue":
    value = line_gest[1]
    if value:
        print("value:", value)
# Additional safety shutdown
if not q.empty():
    item = q.get()
    process.communicate("\n")
    sys.exit()
% \end{lstlisting}
```

Figure 12: Robot control using leap motion data - code snipet part2.

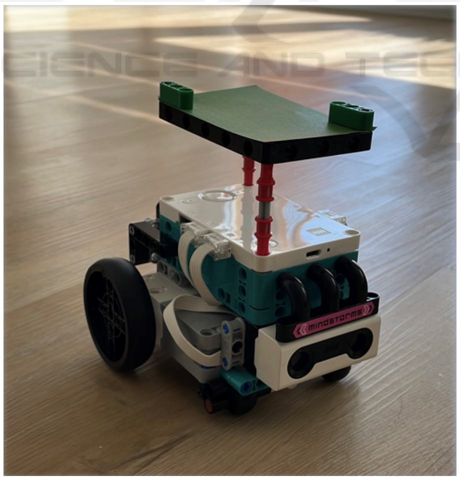

Figure 13: The robot used in the experimental part - based on Smart Element Hub cube with LED screen of the Robot Inventor kit.

possible experimental extensions that we use in practice with other libraries. First and foremost, in the lego set under consideration we have the ability to control servo motors, determine direction of movement, track coloured objects and measure distance to obstacles. Which, in a student learning environment, provides a simple way to test localization filters using distance in map fields and distinguishable colour shades on the floor. Another area of application is the implementation and testing of PID controllers for the precise movement of robots on a map between predefined path co-ordinates, and thus also the testing of path planning algorithms such as D* Lite. Using motor speed control and information about the robot's position relative to the target direction between points. In addition, it is possible to test the algorithms with limited knowledge of the surroundings, e.g. by tracking obstacles encountered and mapping the terrain. In fact, it is only in the field of autonomous robots that it is possible to learn and practically test dozens of algorithms with this simple robot. Not to mention the myriad applications creating hybrid control systems - where we have only shown one of them in our publication.

3 CONCLUSION & FUTURE WORKS

In this paper we presented our new python library for controlling devices built from the lego robot 5in1 mindstorms kit. That is, a toolkit for testing robotics and AI algorithms in constructed within the capabilities of this technology. We reviewed possible applications of the library in education and research. As a case study, we made a presentation of the combination of a Leap Motion device for gesture control of an example robot from the kit. As a demonstrator, we created a fully functional gesture control system for a selected robot. In this work, we offer a highly functional tool for prototyping research scenarios related to the application of robotics and AI techniques in the IoT area. And we demonstrate an in-house tool for learning to develop algorithms - with observation of how they work in practice. In future work, we plan to demonstrate the possibility of using our library with other controllers, e.g. the cyberith virtualiser and the xtion-pro controller.

ACKNOWLEDGEMENTS

This work has been supported by the grant from Ministry of Science and Higher Education of the Republic of Poland under the project number 23.610.007-110

REFERENCES

Anderson, C. A. and Dill, K. E. (2000). Video games and aggressive thoughts, feelings, and behavior in the laboratory and in life. *Journal of Personality and Social Psychology*, 78(4):772–790.

Araujo, H., Mousavi, M. R., and Varshosaz, M. (2023). Testing, validation, and verification of robotic and autonomous systems: A systematic review. *ACM Trans. Softw. Eng. Methodol.*, 32(2).

Cagiltay, K., Kara, N., and Cigdem, C. (2014). *Smart Toy Based Learning*, pages 703–711.

Chacón, A., Ponsa, P., and Angulo, C. (2021). Usability study through a human-robot collaborative workspace experience. *Designs*, 5(2).

Cybowski, W. (2023). Library for smart element hub cube lego robot inventor kit. https://github.com/wcyb/le_mind_controller.

Folk, M. (1981). Review of "mindstorms: Children, computers, and powerful ideas by seymour papert", basic books: New york, 1980. *SIGCUE Outlook*, 15(1):23–24.

Lawitzky, M., Hernández, J. R. M., and Hirche, S. (2013). *Rapid Prototyping of Planning, Learning and Control in Physical Human-Robot Interaction*, pages 73–88. Springer International Publishing, Heidelberg.

LEGO Group (2023). Robot inventor 5-in-1 mindstorms. https://www.lego.com/. Accessed on November 14, 2023.

Malinowski, J. and Artiemjew, P. (2023). Demonstration of a gesture control project using leap motion with the lego robot inventor kit 5in1 mobile robot. https://www.youtube.com/shorts/AjKu5T4IbkE. Accessed on November 14, 2023.

Matthews, P. and Greenspan, S. (2020). *Automation and Collaborative Robotics: A Guide to the Future of Work*.

Miglino, O., Hautop, H., and Cardaci, M. (1999). Robotics as an educational tool. *Journal of Interactive Learning Research*, 10.

Żmudziński, L., Augustyniak, A., and Artiemjew, P. (2016). Control of mindstorms nxt robot using xtion pro camera skeletal tracking. *Technical Sciences*, 19(1):71–81.

Neto, P., Simão, M., Mendes, N., et al. (2019). Gesture-based human-robot interaction for human assistance in manufacturing. *International Journal of Advanced Manufacturing Technology*, 101:119–135.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., and Kafai, Y. (2009). Scratch: Programming for all. *Commun. ACM*, 52:60–67.

Rex Hartson, H. (1998). Human–computer interaction: Interdisciplinary roots and trends. *Journal of Systems and Software*, 43(2):103–118.

Szpakowska, A., Artiemjew, P., and Cybowski, W. (2023). Navigational strategies for mobile robots using rough mereological potential fields and weighted distance to goal. In Campagner, A., Urs Lenz, O., Xia, S., Ślęzak, D., Wąs, J., and Yao, J., editors, *Rough Sets*, volume 14481 of *Lecture Notes in Computer Science*. Springer, Cham.

Ultraleap (2023). Leap motion. https://www.ultraleap.com/. Accessed on November 14, 2023.

UWM (2024). Intelligent robotics laboratory. http://www.uwm.edu.pl/nkr/. Accessed on March 22, 2023.

Zhang, Y. and Doyle, T. (2023). Integrating intention-based systems in human-robot interaction: a scoping review of sensors, algorithms, and trust. *Frontiers in Robotics and AI*, 10:1233328.