# RINS: project report

Blaž Bone[1], Nace Kovačič[1]

[1]Fakulteta za računalništvo in informatiko, Univerza v Ljubljani

## 1 Introduction

The project we undertook as part of the 2022/23 Development of Intelligent Systems competition revolved around a challenging task of developing a robotic system referred to as the RoboSheriff. Our team was charged with designing, implementing, and testing a robot capable of interacting within a simulated environment stylized as a ”Small Wild West City” scene. This was an extensive project with diverse objectives, encapsulating the competencies developed during the first two tasks and pushing the boundaries even further in the field of robotics and intelligent systems.

The core challenge was to enable the robot to identify and locate a “robber” within the simulated environment, a task that entailed several sub-objectives. The robot was required to recognize different entities within the scene including human faces, ”Wanted” posters, and cylinders of different colors. It also needed to identify specific parking slots, indicated by rings of different sizes and colors. The enviroment also included objects that could have been misinterpreted, making it necessary to use as robust methods as possible.

Once the entities were recognized, the robot was tasked to perform a series of actions in a particular sequence. It had to locate all the persons in the city, identify from the posters which robber it was supposed to apprehend, engage in a simple dialogue with the persons to gather information on the potential location of the robber, and search the top of the buildings for the robber. Upon locating the robber, the robot was required to ”imprison” him by taking him to a specific building identified as the prison, and finally park itself in front of this building.

This project presented an opportunity for our team to apply a range of methods and techniques within the realm of intelligent systems, such as face recognition, image detection, autonomous navigation, and simple dialogue implementation. The techniques used were also taught in Machine Perception and Intelligent Systems class, combining the subjects beautifully.

It also posed an array of challenges relating to the integration of different components within the Robot Operating System (ROS). Through this report, we present a detailed description of our approach, the methods applied, and the overall performance of the system that we developed for the third competition.

## 2 Methods

### 2.1 Face detection

The face recognition process in our code is primarily handled through a Deep Neural Network (DNN) using OpenCV. We used a pre-trained DNN model that is known for its performance in

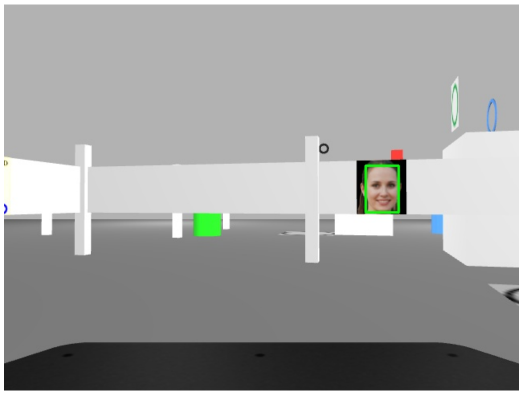


Figure 1: Face detected 1

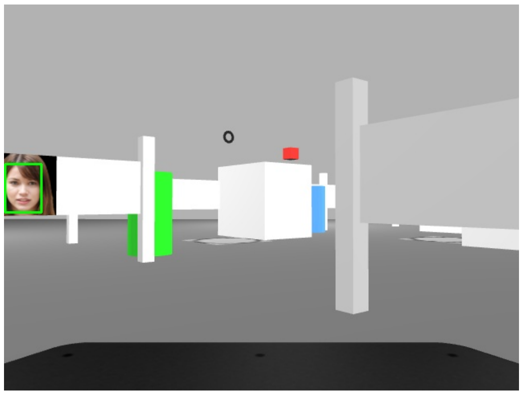


Figure 2: Face detected 2

image recognition tasks. These models have been trained on large datasets of faces, enabling them to recognize a wide variety of faces in different lighting conditions, orientations, and expressions.

In our code, we use the function `cv2.dnn.blobFromImage` to prepare the image for input into the DNN. This function resizes and normalizes the image. We resize the image to 300x300 pixels and perform normalization by subtracting mean RGB values to reduce illumination differences in the image. This blob is then fed into the DNN using `self.face_net.setInput(blob)`.

The network analyzes the blob and returns detections using `self.face_net.forward()`. Each detection includes a bounding box, which represents the coordinates of the detected face in the image, as well as a confidence score indicating the certainty of the detection.

For each detection, we filter out ones with a confidence score lower than 0.50 to reduce false positives. Additionally, we ensure that the detected face's aspect ratio falls within an expected range, which helps us disregard detections that don't resemble a typical face shape or faces that were seen from too narrow of an angle.

Our face recognition system doesn't stop at detecting faces in the current frame. We also maintain a history of faces we have encountered. When we detect a face, we calculate its position in 3D space using depth information from the depth camera. We then match this new detection against previously seen faces based on their positions in the map. If the detected face is deemed to be a new face (i.e., it doesn't match any previously seen faces within a certain threshold), we add the face to our history with its associated data. If it matches a previously seen face, we update the corresponding historical data with the new detection's data.

In conclusion, our face recognition system combines a DNN for image-based face detection with depth information for tracking faces in 3D space over time. This multimodal approach allows for a more robust and versatile face recognition system.

## 2.2 Clyinder detections

Our approach to detecting cylinders in an image is multi-staged, combining image processing techniques with object shape characteristics.

We start by converting the image from the standard RGB color space to HSV (Hue, Saturation, Value). This is more similar to how our human vision perceives color-making attributes.

Following this, we employ a technique called color masking. Color masks are binary images of the same size as our original image. Each pixel of the mask is set to either black or white depending on whether its corresponding pixel in the original image matches a certain color range. In our case, we define the color ranges for red, green, blue, and yellow (colors of cylinders) in the HSV space.

For each detected color, we create a mask, identifying the regions in the image that are of that particular color. We calculate the percentage of the total pixels that each color occupies. The color with the highest occupancy percentage is retained for further processing.

To ensure that we are detecting cylinders, we consider the spatial distribution of the color in the image. A standing cylinder, from a certain perspective, will appear more at the bottom of an image than the top. We create a ”progressive mask” that prioritizes the lower part (cylinders are always on the ground) of the image based on the percentage of color occupancy. We then apply this mask to our image. This gives us a bottom-masked color mask. If there are any non-zero pixels in this bottom-masked color mask, we proceed with the assumption that the color comes from a cylindrical object.

Once we are confident that we have a potential cylinder, we find the contours of the detected color region. Contours are simply the boundaries of an object. Once we have the contours, we approximate them to a polygonal curve. This is done to reduce the complexity of the contour while preserving its structural shape.

A typical characteristic of a cylindrical object is that it will appear as a rectangle in an image when viewed from the side or from afar. A rectangle can be represented as a four-sided polygon in two dimensions. Hence, if our approximated polygon has four sides, and the angles are close to 90 degrees, we have a good reason to believe that the object is cylindrical. This assumption holds especially true because there are no real rectangles on the map.

If a detection passes all the checks, we calculate the centroid, which will be used as the detected location of the cylinder. We also calculate the depth of the detected cylinder in the scene, which can provide information about the cylinder’s distance from the camera and its space on the map.

In conclusion, we leverage color information, spatial distribution, and geometric properties to detect and localize cylinders in an image. This method is a robust approach, allowing us to recognize cylinders of different colors in various scenarios.

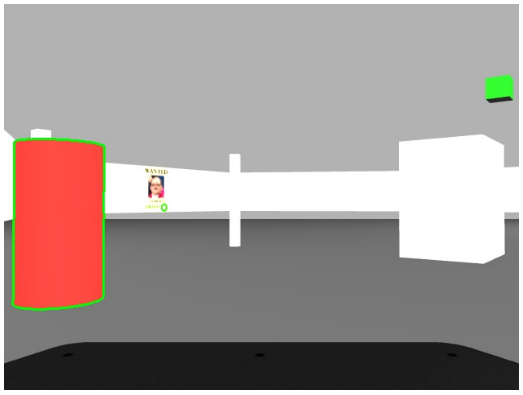


Figure 3: Cylinder detected - Image 1

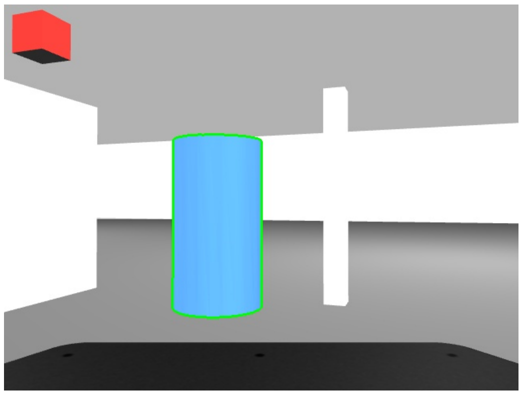


Figure 4: Cylinder detected - Image 2

Figure 5: Cylinder detection results

## 2.3 Ring detection

Our ring detection approach combines various techniques to accurately identify and locate rings in the scene. The process begins by retrieving an RGB image and converting it to grayscale. Histogram equalization is applied to enhance contrast. Adaptive thresholding is then performed to create a binary image.

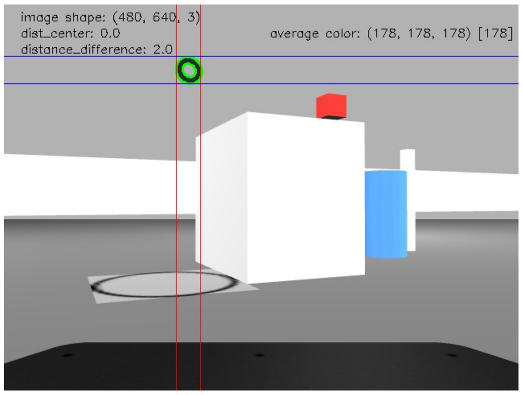
Contours are extracted from the binary image, representing the boundaries of the detected objects. Ellipses are fitted to these contours, and candidate concentric ellipse pairs are identified based on their proximity.

To validate a candidate pair as a ring, several checks are performed, including distance differences and average color range. If a candidate pair passes these checks, further calculations are conducted.

The larger ellipse's size and center are determined, and a region of interest is extracted from the original color image. The average color within the region of interest is extracted using a mask. K-means clustering is applied to the region of interest to extract the main colors present in the region.

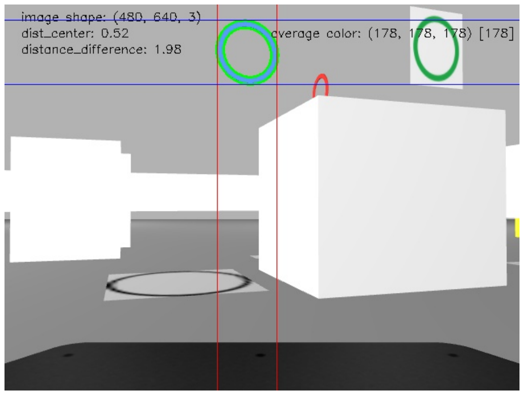
Using depth information, the pose of the detected ring and associated greeting position are obtained. The location and greeting position data are stored in a dictionary, categorized by the ring's main color. Existing data is updated if rings of the same color are detected.

Our approach leverages image processing, ellipse fitting, depth analysis, and color recognition techniques to achieve robust ring detection. By combining information from the color image and depth image, our method enables accurate detection even in complex scenarios.



```
image shape: (480, 640, 3)
dist_center: 0.0
distance_difference: 2.0
average color: (178, 178, 178) [178]
```

Figure 6: Ring detected - Image 1



```
image shape: (480, 640, 3)
dist_center: 0.52
distance_difference: 1.98
average color: (178, 178, 178) [178]
```

Figure 7: Ring detected - Image 2

Figure 8: Ring detection results

## 2.4 Text detection and recognition

Our text detection and recognition process utilizes libraries such as `pytesseract` and `EasyOCR` to perform text recognition.

In the `is_poster` function, an RGB image is captured from the camera. The image is processed using the `pytesseract` and `EasyOCR` libraries to extract text. The extracted text is then passed to the `extract_information` function to obtain reward and color information.

In the `extract_information` function, the input text is processed to extract numerical information and identify the dominant color. The text is cleaned by removing spaces and empty lines. Numerical values are extracted by filtering out non-digit characters and converting the resulting strings to integers. The highest numerical value is considered the reward. The function also identifies the dominant color by searching for specific color keywords in the text.

Our approach combines text recognition algorithms and preprocessing techniques to accurately extract information from the image. The use of multiple text recognition libraries enhances the accuracy and reliability of the information extraction process.

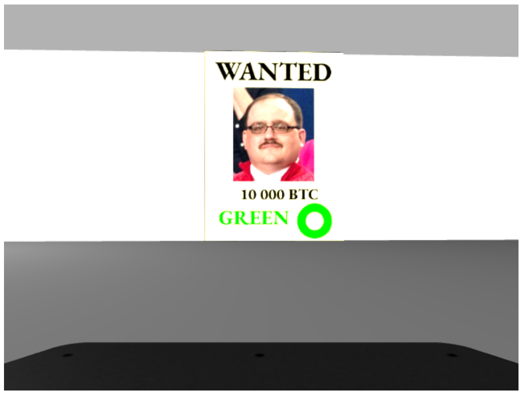


Figure 9: Correctly recognized prison and reward - Image 1

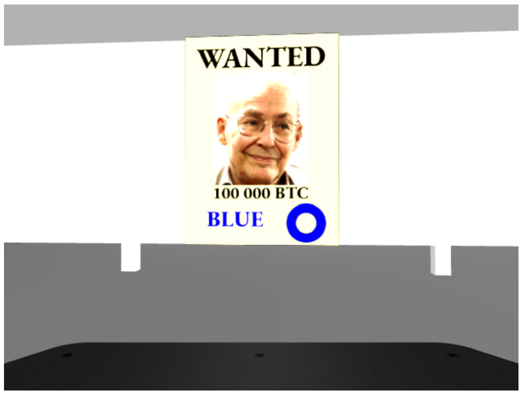


Figure 10: Correctly recognized prison and reward - Image 2

Figure 11: Poster recognition results

## 2.5 Speech Recognition

Our speech recognition module enables the robot to interact with people on the map using spoken language. It utilizes the SpeechRecognition library to convert speech into text.

The robot initiates a conversation by asking the person, "Do you know where the robber is?" Using a microphone as the audio source, the person's response is captured and processed.

The recognized text is then analyzed, searching for specific color keywords related to the robber's location. If any colors are found, they are recorded as possible locations.

This speech recognition module allows for effective communication between the robot and humans, enabling the robot to gather information about the robber's whereabouts.

## 2.6 Movement

In our robot's movement, it is important to note that we have not implemented autonomous navigation. Instead, we have predefined and hard-coded positions that ensure thorough exploration of each part of the map. These positions are strategically mapped to avoid unnecessary spinning or redundant movements, resulting in a more confident and efficient exploration process. By carefully planning the robot's movement, we can navigate the map in a systematic manner, covering all the required areas without unnecessary backtracking or wasted motion. This approach enhances the overall performance of our robot and gives the impression of a confident and purposeful exploration process.

## 2.7 Parking

The parking functionality of our system allows the robot to locate and position itself in front of specific parking slot (prison) indicated by rings of different sizes and colors. The implementation involves image analysis and motion control.

To initiate parking, the robot captures an image using its arm camera and converts it to a binary image. The robot analyzes the binary image to calculate the rotation angle needed to align with the parking slot. It adjusts its position and rotation iteratively based on the highest black pixel's position in the image.

Our parking functionality combines image analysis, depth perception, and motion control to achieve accurate positioning in the middle of the desired parking slot.

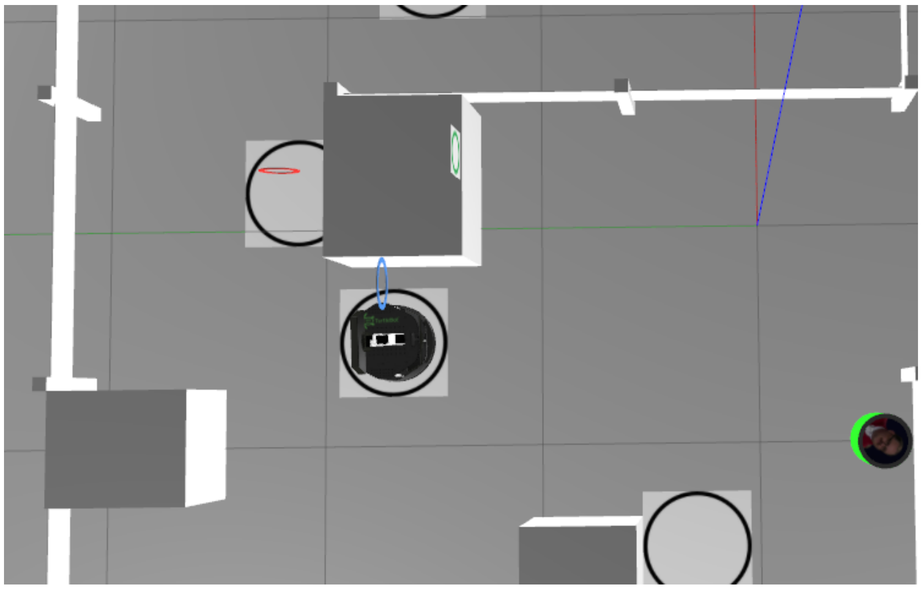
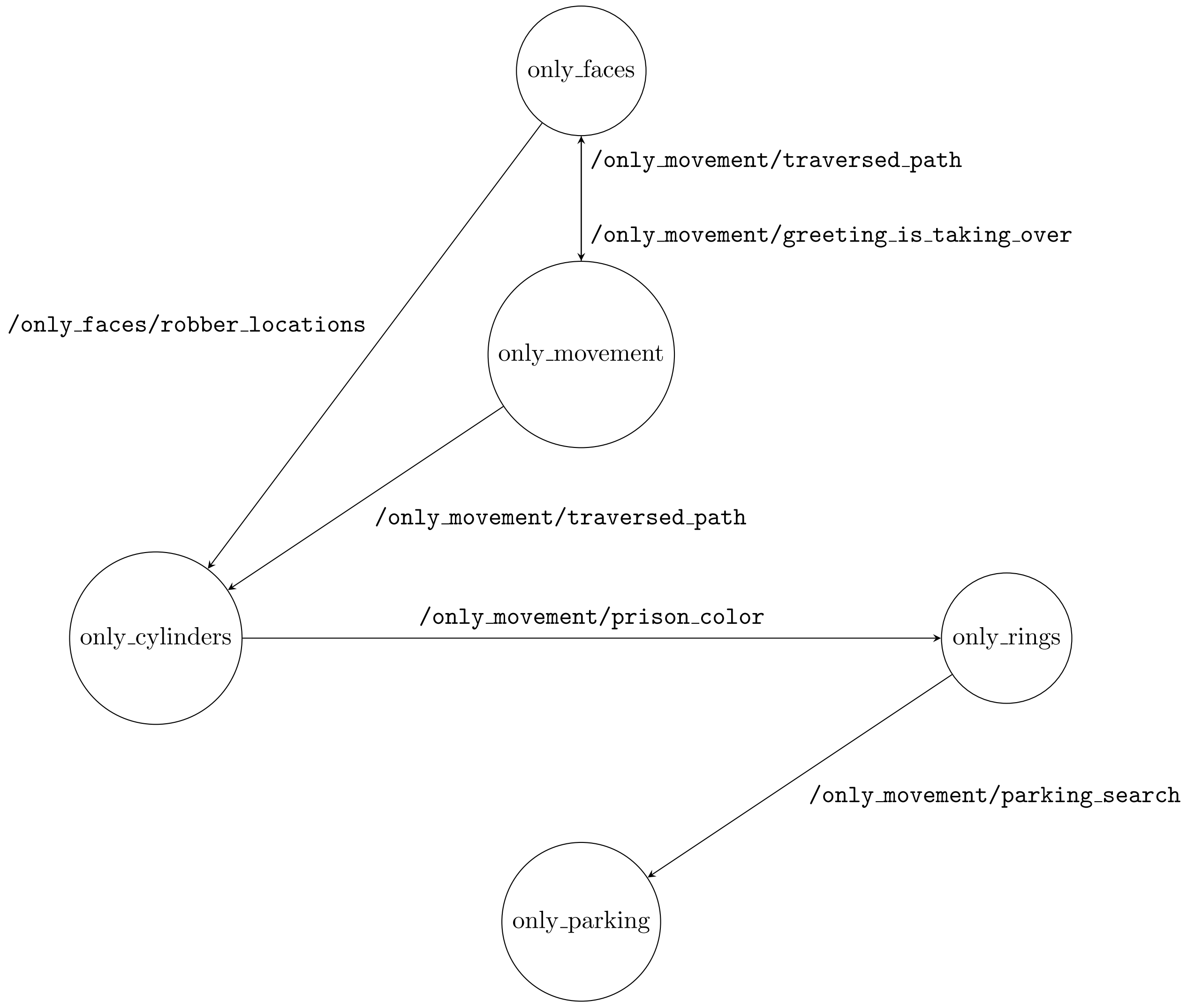


Figure 12: Robot parked in the middle of parking space

## 3 Implementation and integration


only_faces
/only_movement/traversed_path
/only_movement/greeting_is_taking_over
/only_faces/robber_locations
only_movement
/only_movement/traversed_path
only_cylinders
/only_movement/prison_color
only_rings
/only_movement/parking_search
only_parking

### 3.1 Communication when face is detected

While traversing the path, the robot should visit all of the people on the map and ask around for the robber's location. The `only_faces` node is responsible for this task. Once the face has been detected, it publishes to the `only_movement` node that the `only_faces` node will take over navigation of the robot by setting the greeting position of the face as the new move goal. Meanwhile, the `only_movement` node receives a signal that the greeting is taking over via the aforementioned `/only_movement/greeting_is_taking_over` topic. While waiting for the confirmation that the greeting is no longer taking over (sent by the same node, with the message as `False`), the `only_movement` node decrements its path index, ensuring that the robot visits the previous point in the path to ensure proper exploration of the entire map.

### 3.2 Communication once the path is traversed

The `only_movement` node signals the `only_faces` and `only_cylinders` nodes when the robot has finished traversing its predefined path. At this stage, the robot should have detected potential hiding places for the robber (cylinders of two different colors) as well as the color of the ring (referred to as the "prison" color). The communication occurs via the `only_movement/traversed_path` topic, using a `Bool` message. This topic serves as a flag to indicate the completion of map exploration to other nodes.

Upon receiving the completion signal, the `only_faces` node sends a message to the `only_cylinders` node through the `/only_faces/robber_locations` topic. The message contains the color names of the potential hideouts for the robber (e.g., 'green red'). Subsequently, the `only_cylinders` node takes control of the navigation, directing the robot to visit each potential hideout and determine the robber's actual hiding place. Once the last cylinder is visited, the `only_cylinders` node sends a signal to the `only_rings` node, publishing the color of the ring under which the robot should park.

The `only_rings` node acts as an intermediary communication layer between the `only_cylinders` and `only_parking` nodes. It waits to receive confirmation that the robber has been caught by listening to the `/only_movement/prison_color` topic and then sends the appropriate pose to the `only_parking` node, guiding the robot towards the final parking space (referred to as the prison).

## 4 Results

During the competition, our robot exhibited exceptional performance, successfully completing all assigned tasks with high accuracy and efficiency. The individual modules of our system achieved the following results:

- The face detection module accurately recognized and tracked faces within the simulated environment.
- The cylinder detection module effectively identified and located cylinders of different colors.
- The ring detection module achieved accurate detection and localization of rings of different colors.
- The text recognition module accurately extracted information from posters, including reward values and prison colors.
- The speech recognition module enabled effective interaction between the robot and humans, facilitating the gathering of information about the robber's whereabouts.
- The parking module demonstrated reliable performance in navigating and positioning the robot in designated parking spots.

Overall, our robot's exceptional performance showcased the capabilities of our intelligent robotic system, with precise object detection, reliable localization, and successful task completion.

In our system, markers in the form of orange cubes are used for face detection, colored spheres for ring detection, and colored cylinders for cylinder detection. The presence of white spots indicates

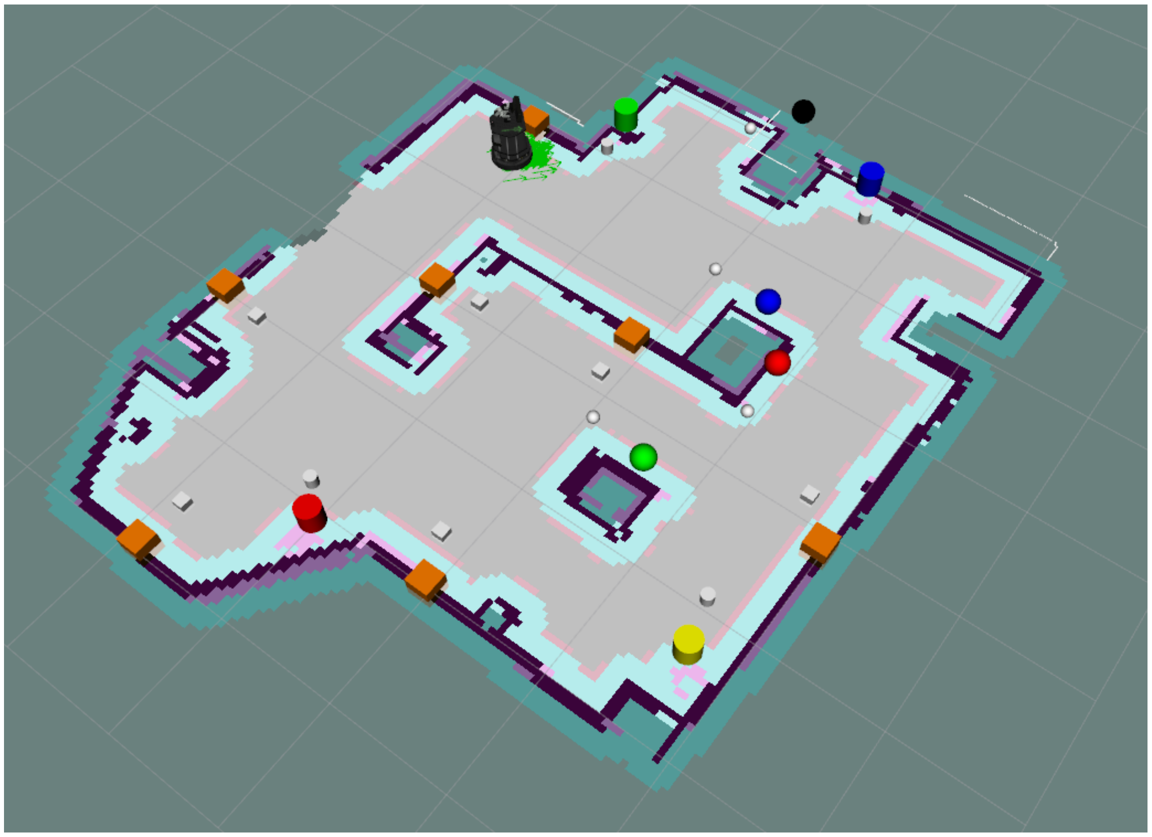


Figure 13: Robot parked in the middle of parking space

the robot's greeting positions, where it approaches and centers the detected objects in its field of view. We successfully detected all objects, and our markers are dynamically updated during the run, providing more reliable results by averaging the positions of the captured objects instead of relying solely on the initial detection.

## 5 Division of work

Blaž and Nace (collaborative work): We worked collaboratively on all the tasks listed above, utilizing pair programming methodology. This approach allowed us to share ideas, exchange knowledge, and jointly implement the different functionalities of the RoboSheriff system. We divided the work equally, with each of us contributing approximately 50% to the overall project. Together, we successfully developed and integrated face detection, poster detection, ring detection, color recognition, approaching faces, parking, digit recognition, circle and color name recognition, cylinder detection, approaching cylinders, and dialogue with ASR into the system. Our joint effort ensured a balanced distribution of tasks and maximized the efficiency of our development process.

## 6 Conclusion

In this project, we have developed a robotic system called RoboSheriff capable of interacting within a simulated environment and performing tasks such as face detection, cylinder detection, ring detection, text recognition, speech recognition, parking, and fine maneuvering. Our system integrates various components and leverages intelligent system techniques to achieve its objectives.

Our implementation was successful, as we have completed all of the required objectives.

Regarding hardware and software challenges, we encountered limitations due to the use of Ubuntu 20.04, which is a 3-year-old distribution with outdated software. This affected the availability of the latest features and improvements. Additionally, the performance of our system heavily relied on the computer's capabilities, requiring a powerful machine to run multiple neural networks for tasks like face recognition and OCR. Different hardware setups could impact the system's performance, with more powerful computers achieving higher frame rates and better overall performance.

Furthermore, reproducing the results proved challenging due to the non-deterministic nature of ROS. The behavior of the system could vary across different runs, making it difficult to precisely reproduce the exact results in each execution. However, our robust approach managed to consistently perform correctly every time we presented the task to the professor.

In conclusion, our project successfully developed the RoboSheriff robotic system, integrating various intelligent system techniques to perform tasks within a simulated environment. We have gained valuable insights into face detection, cylinder detection, ring detection, text recognition, and speech recognition. Future work could involve exploring more advanced algorithms and further optimizing the system's performance.