Autonomous mobile robot for automatic out of stock detection in a supermarket

Giuseppe De Simone
University of Salerno
gidesimone@unisa.it

Pasquale Foggia
University of Salerno
pfoggia@unisa.it

Alessia Saggese
University of Salerno
asaggese@unisa.it

Mario Vento
University of Salerno
mvento@unisa.it

Abstract

Out of stock is among the main causes of sales losses for retailers. In order to face this issue, in this paper we propose ROSCH (the RObot for SCHelves analysis), an autonomous mobile robotic platform based on ROS framework whose aim is to inform the human operators in case of empty or partially empty shelves, so as to speed up the refilling process. ROSCH is able to autonomously move inside an environment, and to autonomously identify those shelves which are empty or partially empty, thanks to the use of a deep learning based detector, validated on a dataset composed by about 2000 manually annotated images, 900 of them acquired by our team in three different supermarkets in Italy. The proposed system has been tested in a supermarket in Salerno (Italy) at working time; the analysis conducted demonstrates that the proposed system is able to reliably support the supermarket staff, being 8 times faster than the human operator in its common manual out of stock detection activity.

1. INTRODUCTION

Nowadays, the demand for quick and efficient grocery shopping experiences is increasing, and supermarkets are continuously exploring new ways to improve this experience for their customers. Indeed, a challenge that customers often face with is the unavailability of their desired products on the shelves (the so called Out of Stock - OOS).

Facing an OOS implies in 9% of the cases not to buy at all that product, meaning that the intended purchases are definitively lost. The Harvard Business Review [1] also demonstrates that this abandoned purchase translates into a sales loss of 4% for a retailer, meaning for retailer giants about 40 million dollars of loss per year.

Starting from this consideration, in recent years there has been growing interest in developing video analytic algorithms that can automatically identify empty shelves in a supermarket. In [14] the authors use a depth camera with a top-down view of the shelves being monitored. They create a model of the shelf during a preliminary calibration process and use the 3D point cloud to estimate the percentage of the shelf occupied by items. The main drawback of this approach is that it is limited to countertop shelves, refrigerated counters, or similar shelves that can be monitored with a top-down view and requires a preliminary calibration, meaning that the camera must be fixed.

In [19], a deep learning-based detector using YOLOv4 was proposed to identify the three categories of *Product*, *Empty Shelf*, and *Almost Empty Shelf*. In [5], a Faster R-CNN was utilized to determine the position of products and unknown areas within the shelves were detected using the Canny Operator. Indeed, less edge means that the regions inside the shelves can be considered OOS. In [3], the authors installed a camera on a shopping cart and formalized the problem of OOS as a segmentation problem, using a U-Net architecture for distinguishing empty shelves from everything else in a supermarket.

The authors in [10] have been among the first ones to explore temporal information by using videos instead of still images. The authors focused on changed regions through background subtraction and moving object removal algorithms, then employed a CNN to classify *Products Taken* and *Products Returned*, or *Slight Changes Due to a Shopper Touching It*. However, this type of algorithm still requires a fixed mounted camera.

Using fixed cameras would imply to spread a quite huge amount of cameras in the supermarket (the bigger is the supermarket, the higher will be the number of cameras to be installed), so as to cover all the shelves. Vice-versa, the implementation of autonomous mobile robots equipped with a camera and a video analytic algorithm in charge of autonomously monitoring the status of the shelves provides a cost-effective solution for detecting empty shelves and reducing the financial losses associated with out-of-stock items. Indeed, the use of real-time information on the availability of products, combined with the low cost of this kind of solution, will allow supermarkets to restock empty

shelves promptly and provide a better shopping experience for their customers.

In [11] a first attempt in this direction is made: a set of robots, properly equipped with a camera, is able to move in a supermarket, both autonomously over some prefixed paths and manually, through teleoperation. The human operator can thus remotely monitor by video cameras installed on board of the robot and he is facilitated in detecting any OOS. Even if a preliminary work since the detection is manually performed by the operator, the main advantage of the proposed system is that it does not require modifying existing infrastructure of the store, and the cost of the entire solution is cheap.

In [15] the authors propose a mobile robotic platform based on Turtlebot able to autonomously move in the supermarket and detect potentially empty shelf. The navigation framework exploits data coming from carts and baskets tracking data, collected by indoor Ultra-WideBand Localization. Such data are used both for allowing the robot to visit more often the areas in the supermarkets where customers spend more time (which are supposed to suffer more of OOS issue) and also to avoid obstacles. The empty shelf detection is formulated as an image classification problem by employing a deep neural network; three classes are identified: *positive* (images of shelf with products at a special offer), *neutral* (images of planogram in a standard layout. It also includes empty shelves) and *negative* (images of SOOS situations, where most of the shelves are empty). Anyway, the system requires a quite complex and expensive infrastructure to be installed for localization.

In [12] the authors show the importance of a socially aware robot in a supermarket for OOS detection operations. The authors experimentally evaluate that such kind of robot is generally accepted by customers, especially if the robot provides a customer-centered approach, keeping appropriate distances to users while choosing a product.

Within this framework, in this paper we provide a ROS based framework for autonomously controlling a mobile robot inside a supermarket, aiming at autonomously identifying empty shelves. The proposed system is designed to autonomously navigate within the supermarket, ensuring collision avoidance with both objects and people. While moving, the system is responsible for identifying both empty and partially empty shelves. The OOS detection is autonomously performed by means of a deep learning based detector based on YOLOv6, properly validated by using a dataset of about 2000 images, 900 of them acquired by our team in three different supermarkets in Campania region, in the south of Italy. The dataset has been manually labeled with two categories, namely *empty* and *partially empty* shelves.The status of the shelves (together with its position) is finally shared with the supermarket employees. We named the proposed system ROSCH (RObot for SCHelves analysis).

An implementation of the framework based on the Pepper robot has also been provided, and the system has been tested in a supermarket in Salerno, Italy, during operating hours. The system has demonstrated impressive results: about 85% F1-Score for the automatic detection of empty and partially empty shelves (about 92% if we only consider empty shelf detection), and about 92% of navigation tests successfully completed inside the supermarket. Moreover, in order to highlight the efficiency of the proposed system, a comparison between its performance and that of a human operator has been conducted. The results obtained indicate that the system is able to complete a round of the supermarket while detecting out-of-stock items approximately 8 times faster than a human operator.

2. Proposed system architecture

In this section we detail the architecture of ROSCH, the proposed autonomous mobile robotic platform for empty shelf detection. ROSCH software architecture consists of a modular software architecture based on the Robotic Operating System (ROS), a de-facto standard framework for robot programming. This choice is justified by the fact that ROS allows for hardware independence, together with the possibility to easily implement nodes (even with different programming languages) communicating according to the publisher/subscriber protocol [6].

The architecture of the proposed system is shown in Figure 1. It has been thought to accomplish the following main tasks: (i) navigation, through localization, motion planning and obstacle avoidance; (ii) SLAM, for map building, to be activated only during a startup phase; (iii) people and empty shelves detection through visual analysis. More details about each of the above mentioned modules is shown in the following of this section.

2.1. Hardware setup

The robotic platform we decided to use is the humanoid robot Pepper. This choice is motivated by the fact that Pepper could be potentially used inside a supermarket not only for empty shelves detection, but also for social interactions with the customers..

Pepper is equipped with an Inertial Measurement Unit (IMU) consisting of 3-axis gyroscope capable of measuring angular speeds of approximately 500°/s, along with a 3-axis accelerometer measuring accelerations of approximately 2g. Additionally, the robot is equipped with a set of rotary encoders. Also, it is equipped with three laser sensors located on its base. Each sensor has a field of view of 60 degrees, and the total number of data points obtained by the laser sensors is 45 (15 per side). In order to enhance the accuracy of the localization algorithm, we have also exploited the data from the depth camera ASUS Xtion 3D, located in

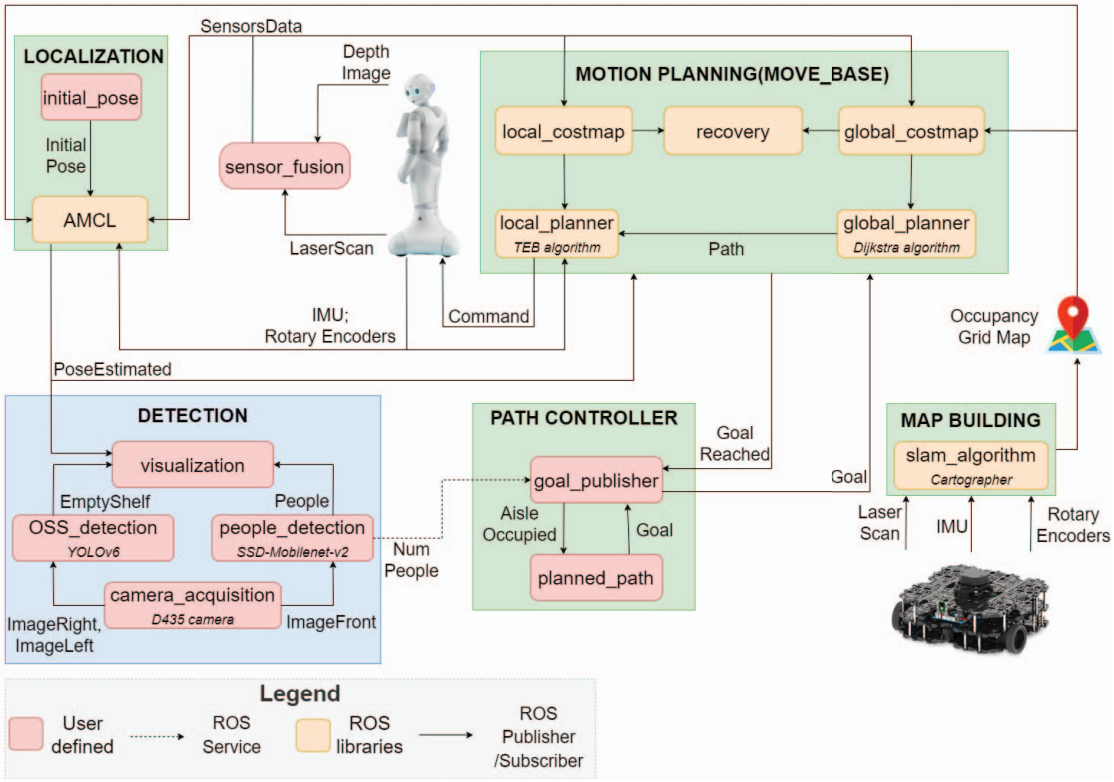


Figure 1: ROS architecture of the proposed ROSCH system. The green areas describe the navigation modules employed in the system, including localization, motion planning and path controller. The module denoted as **map building** employs data from the Turtlebot's sensors to construct a representation of the environment. Subsequently, the **localization** module utilizes sensor data, specifically laser scan and depth images, to determine the robot's position within the environment. The **motion planning** module generates coherent paths for the robot to follow in order to reach its assigned waypoint using the *move base* package by ROS. The **path controller** selects the relevant areas of the map to be explored and subsequently dynamically generates the corresponding waypoints. The blue area includes computer vision based tasks, and in particular it consists of a **camera acquisition** node, responsible for the acquisition of video streams from the cameras. The acquired video streams are then processed by an **out-of-stock detection** algorithm, and by a **people detection** one. The **visualization** node shows the outputs of the detection algorithms for the user, providing a visual representation of the detected out-of-stock items and people.

the head of the robot. Indeed, the point clouds generated by the three laser sensors, as well as the data obtained from the camera, have been fused together to produce a more comprehensive representation of the environment. For the analysis of the scene pertaining to both people and empty shelf detection, an ensemble of three RealSense D435 cameras has been incorporated. These cameras are endowed with both depth and RGB modules, although only the RGB data has been utilized. The RGB sensor exhibits a resolution of 2 MegaPixels and offers a horizontal field of view of 69° and a vertical field of view of 42°. The cameras have been mounted on the head of the Pepper robot using a custom 3D printed support. Two of these cameras take the right and left of the robot, respectively, while the third camera is looking in front of the robot. A visual representation of this camera setup is shown in Figure 2. Finally, Pepper has been also equipped with an NVIDIA Jetson Xavier NX, a low power embedded device where we install both navigation and artificial vision algorithms.

2.2. Map Building

One of the first steps has been the generation of the map of the working environment. This is an important and not negligible step, since a representation of the surrounding environment is mandatory for the robot in order to allow its autonomous movement. This task has been done by a Simultaneous Localization and Mapping (SLAM) algorithm, namely Cartographer [9], and the generated map has been represented as an *Occupancy Grid Map*.

It is important to note that, differently from the operating

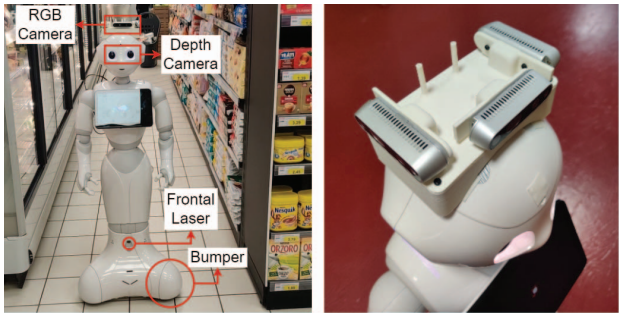


Figure 2: Pepper during the tests at the supermarket. The three cameras have been added on top of the head in order to look frontally, on the right and on the left side, so as to take the shelves while traversing the corridors.

phase, a different robotic platform has been used in order to build a reliable map, due to the blind spots and the low number of data points in Pepper robot, not able to cover 360 degrees. In particular, a Turtlebot 3 robot has been employed. It is equipped with the LDS-01, a 2D laser scanner renowned for its capacity to achieve full 360-degree environmental perception. Furthermore, the robot is outfitted with an Inertial Measurement Unit (IMU) that encompasses a 3-axis Gyroscope and a 3-axis Accelerometer, thereby facilitating accurate orientation and acceleration measurements. Additionally, the inclusion of rotary encoders contributes to the robot's capability to precisely quantify its motion and position. This comprehensive suite of sensors enables the Turtlebot 3 to effectively and reliably capture essential data, culminating in the construction of an intricate and dependable environmental map.

2.3. Autonomous navigation

The development of an autonomous navigation system for a robot requires that the navigation algorithm exhibits robustness in handling unexpected scenarios, such as modifications to the environment and presence of people. This requires the definition of a localization and path planning algorithm, combined with a reliable obstacle avoidance algorithm that guarantees the safety of the robot, as well as the one of any potential objects or individuals in the environment.

Given the map, the localization is achieved by the *Adaptive Monte Carlo Localization* (AMCL) algorithm [18]. It is important to highlight that errors at localization step are automatically managed by the system. Indeed, in case the planners fail to generate a path to reach the assigned goal due to obstructions, the robot automatically initiates a "recovery" behavior by rotating on itself. This behavior is intended to reduce errors in localization and eliminate any spurious measurements of the system's internal map.

Furthermore, our navigation system employs a global planner based on Dijkstra algorithm and a local planner based on a Timed Elastic Band (TEB) algorithm [16]. In order to avoid collisions with both static and dynamic obstacles, a dedicated inner map built on top of the map representation is created by the robot itself. Differently from the occupancy grid map, this representation, dynamically updated by sensors data, takes into account both static and dynamic obstacles. Indeed, during path generation, the global planner utilizes the inner map to generate a collision-free path. Meanwhile, the local planner generates a path that follows the global path while respecting the robot's degree of freedom. In this way, when a new obstacle is detected, it is added to the inner map, allowing the global and local planners to modify previously generated paths and to avoid collisions.

The areas of the map to be explored are specified by the user in the form of a list of waypoints, which are read by the global planner in a sequential manner. Upon receiving the waypoints, the global planner generates a consistent path to reach the assigned goal. It is important to note that the robot has to move inside the supermarket also in presence of customers. This is why, in order to avoid to block the path of the customers during their buying, we also include in our architecture a *path controller*. Indeed, for each waypoint, the system verifies the presence of individuals using a people detection algorithm. This step is performed to ascertain whether an aisle is occupied or unoccupied. Then, if there is enough space to navigate without disrupting the people in the aisle, the system attempts to reach the assigned waypoint without causing any disturbance. On the other hand, if there isn't sufficient space (due to the limited size of the aisle), the waypoint that corresponds to the occupied area is skipped, and the system proceeds to the next one. In this way, the customer can continue buying without any interruption due to the robotic platform.

2.4. People detection

The goal of this module is to detect if there are people into the aisle. In order to avoid system delay, the people detection must be as fast as possible. So to respect the real-time constrain, the people detection is performed using the SSD-Mobilenet-v2 [17]. This choice is justified by the fact that, being a single stage detector, it is able to reliably and efficiently detect the presence of objects of interest. Additionally, to further enhance the inference time, we optimized the network using NVIDIA TensorRT. The detection of people is carried out on demand every time a waypoint is reached, in general, each time that the robot reaches the beginning of the aisle. The number of individuals detected is provided to the *path controller*, it checks if the number of people presented in the aisle is under a predefined threshold that describe the maximum number of persons inside a corridor. In this case there is enough space and the robot can

visit the aisle. On the other hand, the *path controller* selects the next feasible waypoint.

2.5. Empty Shelf detection

The empty shelf detection problem has been formulated in terms of object detection. Two classes have been introduced, namely *empty shelf* and *partially empty shelf*.

Among the detectors available in the literature, anchor-free detectors have proved very promising generalization capabilities, especially with limited data available for training. In particular, we decided to adopt YOLOv6 [13], a single-stage anchor-free object detection model based on the YOLO architecture. YOLOv6 has proved to be 51% faster than anchor-based detectors; also, it has proved to be more accurate thanks to the introduction of the SimOTA dynamic label assignment strategy, to dynamically allocate positive samples: indeed, the number of positive samples is increased through the cross-grid matching strategy, so that the network can quickly converge.

Also, in order to further improve the accuracy and accelerate the network convergence, a SIoU bounding box regression loss is introduced. This loss allows to consider not only the distance between the center points and the aspect ratio, but also introduces the vector angle between the required regressions, which effectively reduces the degree of freedom of the regression and allows for a fast convergence.

Given the limited size of the available training set, data augmentation techniques have been also employed, namely (i) *HSV increasing*: hue, saturation and value components in the image have shown an increase; (ii) *degrees*: the image is rotated by a degree between 0° and 360°; (iii) *scaling*: the image is scaled outward and inward; (iv) *shearing*: images is sheared from both corners in the x and y direction; (v) *translation*: the image is shifted into various areas along the x-axis or y-axis; (vi) *flipping*: the image is flipped horizontally and vertically. In addition to these basic techniques, two strong data augmentation techniques are also applied, namely Mosaic [4] and Mixup [20]. Finally, in order to further speedup the inference time, the network has been optimized with TensorRT framework.

3. Experimental results

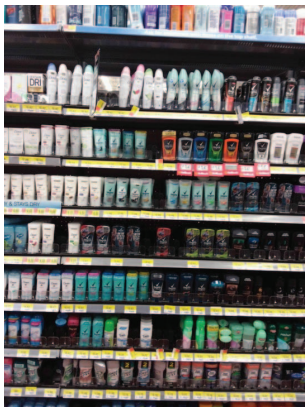
3.1. Empty shelf detection

The dataset is obtained by combining publicly available datasets with a novel dataset collected by our researchers.In terms of publicly available datasets, we combine samples from Grocery Products [7], WebMarket dataset [2] and SKU110K [8].

Grocery Products: the dataset has been made for grocery products detection. The training set is composed by 8350 images, each containing a single grocery product, without any background (namely, it is not inside a shelf) This is why we did not use this part of the dataset. Vice-versa, the test set is made by 680 shelves images; this is what we included in our dataset. An example is shown in Figure 3b.

Dataset	images	objs	AE objs	E objs
Grocery Products [7]	680	700	619	81
Web Market [2]	300	1435	693	742
SKU110K [8]	88	189	67	122
MIVIA Supermarket (Our)	900	2348	1404	944
Total	1968	4672	2783	1889

Table 1: The table reports, for each dataset, the number of images and the number of objects (*objs*)), partitioned into *empty shelf* (E) and *almost empty shelf* (AE). The total number of images and objects used in our experimentation is also reported.



(a) SKU110K

(b) Grocery product

(c) WebMarket

Figure 3: Three examples of images from SKU110K (a), Grocery Products (b) and WebMarket (c) datasets.

WebMarket: the dataset is made by 300 shelves images. As the previous dataset, it has been collected for grocery products detection, thus the labels refer to products and not to empty shelves. An example is shown in Figure 3c.

SKU110K: the dataset is composed by 11743 RGB shelves images. Most of them takes shelves not with a frontal view. Thus, we select 88 frontal images that we included in our dataset. An example is shown in Figure 3a.

All the 1068 images have been manually annotated by our experts. Given the limited size of the dataset, we decided to collect a novel dataset that we call *MIVIA Supermarket dataset*. It is composed by 900 images collected in three supermarkets in three cities in the south of Italy

Figure 4: Sample images from MIVIA Supermarket Dataset.

Figure 5: Two examples of images, together with the overlay of the predicted bounding boxes (in red the empty, in orange the almost empty).

(Napoli, Agropoli and Salerno, respectively). Sample images from Mivia Supermarket dataset are shown in Figure 4. The dataset has been collected so as to have different typologies of shelves, products, view angles, zoom levels and lighting conditions. As the previous ones, also such images have been manually labeled.

Totally, the dataset is composed by 1968 shelves images containing 4672 objects (each one annotated with bounding boxes), partitioned as follows: 2783 almost empty objects and 1889 empty objects. The detailed number of images and objects for each dataset is reported in Table 1.

Given the size of the dataset, a 5-Folds cross validation analysis have been performed. For each test, 80% of the datasets has been considered for training (4 folds), 10% for validation and 10% for testing. The different versions of YOLOv6, namely N/T/S/M/L have been taken into account. The model's performance was evaluated using standard metrics including Precision, Recall, and F1-score. These metrics were computed for each fold of the dataset, and the average results were reported. The model was optimized for efficient inference using the NVIDIA TensorRT SDK, which allowed for faster processing and improved performance.Furthermore, the frame rate, measured on a NVIDIA Jetson Xavier NX board, was determined to assess the system's processing speed. The final results, presented in Table 2, demonstrate that YOLOv6-S achieved the highest accuracy with an average F1-score of 84.61% at a frame rate of 28.78 FPS. Conversely, YOLOv6-N showcased the best frame rate of 78.6 FPS. Considering the slight variance in F1-scores among the models, YOLOv6-T emerged as the most suitable method due to its balanced tradeoff between accuracy and speed.

Some example of the output of the proposed system are reported in Figure 5. We can see that the proposed algorithm performs well in case of different categories of grocery products, different zooms, lighting conditions and different categories of shelves. Other than a quantitative analysis, a qualitative analysis has been also performed. Indeed, we have analyzed the errors of the proposed system and we got that almost 25% of the errors is due to products placed on top of other similar products, with challenging lighting conditions. Some examples are reported in Figure 6(a-b). Furthermore, almost 30% of the errors are due to shelves taken with top or side views; some examples are shown in Figure 6(c-d). This analysis let us understand that the best operating conditions of the proposed system requires a frontal view.

3.2. ROSCH evaluation

The experimentation of the system has been conducted in a supermarket in Salerno, Italy, at working hours, so as to evaluate the reliability of the proposed system even in presence of customers moving inside the supermarket in an uncontrolled way. While acquiring the map by SLAM, the robot has been manually tele-operated by the human operator. The acquired map, used during the autonomous navigation by the robot, is shown in Figure 7.

In order to assess the functionality of the system, a set of waypoints is defined and provided as input to the robot as reference paths. We design five test paths (shown in Figure 8), with the aim to validate the system in the different

Model	Almost Empty			Empty			Average			
	P	R	F	P	R	F	P	R	F	FPS
YOLOv6-N	75.43%	80.33%	77.74%	89.67%	91.63%	90.61%	81.50%	85.13%	83.22%	78.6
YOLOv6-T	78.87%	80.70%	79.76%	88.63%	91.70%	90.18%	83.08%	85.47%	84.25%	43.91
YOLOv6-S	79.51%	79.59%	79.48%	91.05%	91.93%	91.45%	84.46%	84.88%	84.61%	28.78
YOLOv6-M	78.43%	78.67%	78.47%	90.83%	91.92%	91.37%	83.76%	84.31%	83.98%	14.23
YOLOv6-L	78.32%	78.66%	78.44%	89.37%	90.66%	89.97%	83.16%	83.72%	83.39%	8.38

Table 2: Results achieved by the proposed system, in terms of Precision (P), Recall (R) and F1-Score (F)

(a) (b) (c) (d)

Figure 6: Examples of images where some errors have been done by the proposed system. We can see the predicted bounding boxes (in red the empty, in orange the almost empty) and the missed ones (in white). (a,b) show products placed on top of the other ones, with challenging lighting conditions; (c,d) show examples of missed products due to top-side view.

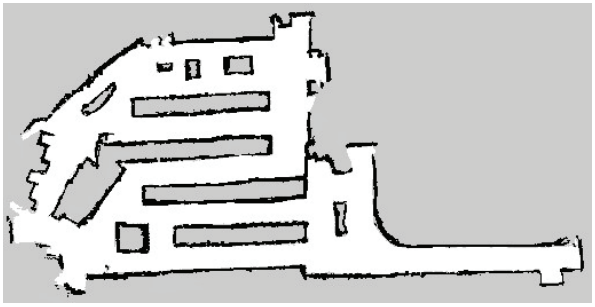


Figure 7: Map of the supermarket environment where our tests have been conducted.

possible conditions.

We consider the following metrics for the evaluation: the time required to complete each successful path (measured in seconds), the number of recovery operations performed by the robot, and the number of collisions with people or objects in the environment. In this context, a collision with an object during the test is considered a failure.

In order to account for the inherent variability in the environment, each path is executed five times. The mean and standard deviation of the completion time for each path are thus computed.

The first path, denoted as *Market Path 1* (Figure 8a), is a quite small route intended to evaluate the robot's navigation and localization capabilities in an unchanged environment (no additional items have been placed with respect to the map acquisition time), even if with some tight curves. The second path, *Market Path 2* (Figure 8b), is designed to test the robot's ability to navigate a longer route that includes a narrow section of the supermarket and some more demanding maneuvers in tight aisles. *Market Path 3* (Figure 8c) evaluates the robot's navigation ability in the right half of the supermarket. This path includes a challenging section between points 2 and 3, due to the changes in the environment from the time in which the map has been built with respect to the testing time. *Market Path 4* (Figure 8d) tests the robot's ability to navigate the aisles in the opposite direction to the previous paths, by covering more or less the whole supermarket. Finally, in *Market Path 5* (Figure 8e), the robot's ability to localize itself and return to its starting point is evaluated by traversing a few points twice.

The results of the experiments are summarized in Table 3. Overall, ROSCH completed successfully 23 paths over 25, with a 92% success rate of completed tests. One possible cause of failure in Market Paths 1 and 5, resulting in 2 collisions, has been the presence of blind spots in the environment, which resulted in inaccurate map updates by the local planner. Furthermore, it was observed that in challenging situations such as in the Market Path 5, the number of required recovery behaviors increases. This resulted in a longer average path completion time and a higher standard deviation. Anyway, the system is still able to successfully perform its task.

(a) Market Path 1 (b) Market Path 2 (c) Market Path 3 (d) Market Path 4 (e) Market Path 5

Figure 8: Experiments carried out inside the supermarket at working ours to evaluate the navigation system. Each experiment has been repeated 5 different times.

Furthermore, in order to evaluate the automatic re-planning capability of the system in presence of persons, we also perform a second set of experiments aiming at evaluating the system's ability to accurately identify occupied aisles and avoid them in the presence of people, in case re-planning automatically the route. It is important to note that in this specific supermarket, due to the limited size of the aisle, there is not enough space for both the robot and the person. Thus, the number of persons requiring a re-planning of the path due to occupied aisle has been set to one. It implies that a test is considered completed if the system detects the presence of at least one person occupying an aisle and skips visiting it, but successfully completing the rest of the path planned. The test paths are shown in Figure 9. The presence of people in the images is represented by a red circle. To assess the system's performance, the designated paths were executed five times, resulting in a total of fifteen trials. Test *People Path 1* (Figure 9a) analyzed the robot's behaviour when an aisle is occupied by only one person. Test *People Path 2* (Figure 9b) evaluates the system's performance when an aisle is occupied by two people. In test *People Path 3* (Figure 9c), the system has been tested in a more complex scenario, where the first aisle is occupied by two people, so the robot is expected to navigate to the third aisle and stop before visiting it because it is occupied by one person. The results indicate that the system passed all the tests 5/5 times.

A final test has been performed in order to compare the performance of the ROSCH proposed system with respect to the human operator. We asked the human in charge of the shelves inside the supermarket to move inside the the supermarket (excluding the checkout area and the cured meat area, where the proposed system has no sense) and count OOSs. Thus, we measure the time required by the human to do it manually; then, we measured the time taken by the robotic platform to complete a round of the supermarket. Finally, we compared the two measured times: the robot successfully completed its task in about 7 minutes. Vice-versa, the human required a bit less than 1 hour, meaning 8 times more time than the automatic proposed robotic system. This impressive result confirms the efficiency and the effectiveness of the proposed system.

Path ID	Time (s)	#Recov	#Coll
Market Path 1	119.00 ± 1.87	0	1
Market Path 2	223.80 ± 17.11	1	0
Market Path 3	129.00 ± 6.75	1	0
Market Path 4	251.20 ± 12.30	2	0
Market Path 5	258.75 ± 18.71	3	1

Table 3: Results of the *Market Path* tests conducted, reporting the average time (and the standard deviation) taken to complete the task in seconds **(Time)**, the number of recoveries **(#Recov)**, and the number of collisions **(#Coll)**.

A demo of the system in action is shown at the following link: https://youtu.be/h_g1oAEbsx4.

4. CONCLUSIONS

In this paper we propose ROSCH, an autonomous mobile robotic platform able to automatically detect out of stock in

(a) People Path 1 (b) People Path 2 (c) People Path 3

Figure 9: Experiments conducted in presence of people. The red point represents the position of the person.

a supermarket. ROSCH is based on ROS framework; it is able to autonomously navigate inside the supermarket, and it has been extended with advanced artificial vision capabilities based on deep learning for detecting people, with the aim to avoid interrupting customers buying process, and also for empty and partially empty shelf detection, with the aim to autonomously inform the staff about missing products to be immediately refilled. ROSCH has been tested in a supermarket in Salerno, showing impressive capabilities in both navigation and OOS detection tasks. Future works include the possibility to further extend ROSCH with social capabilities, so as to also interact with the customers proposing additional products that they could buy.

ACKNOWLEDGMENT

This research has been partially supported by A.I.Tech (www.aitech.vision) and A.I. Ready, a spin-off company of the University of Salerno.

References

[1] Harvard business review: Stock-outs cause walkouts. https://hbr.org/2004/05/stock-outs-cause-walkouts, 2004. Accessed: 2022-07-01. 1

[2] Webmarket dataset. https://www.kaggle.com/datasets/manikchitralwar/webmarket-dataset, 2021. Accessed: 2022-07-13. 5

[3] Dario Allegra, Mattia Litrico, Maria Ausilia Napoli Spatafora, Filippo Stanco, and Giovanni Maria Farinella. Exploiting egocentric vision on shopping cart for out-of-stock detection in retail environments. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 1735–1740, 2021. 1

[4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020. 5

[5] Jun Chen, Shu-Lin Wang, and Hong-Li Lin. Out-of-stock detection based on deep learning. In De-Shuang Huang, Vitoantonio Bevilacqua, and Prashan Premaratne, editors, *Intelligent Computing Theories and Application*, pages 228–237, Cham, 2019. Springer International Publishing. 1

[6] Pasquale Foggia, Antonio Greco, Antonio Roberto, Alessia Saggese, and Mario Vento. A social robot architecture for personalized real-time human-robot interaction. *IEEE Internet of Things Journal*, pages 1–1, 2023. 2

[7] Marian George and Christian Floerkemeier. Recognizing products: A per-exemplar multi-label image classification approach. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 440–455, Cham, 2014. Springer International Publishing. 5

[8] Eran Goldman, Roei Herzig, Aviv Eisenschtat, Jacob Goldberger, and Tal Hassner. Precise detection in densely packed scenes. In *Proc. Conf. Comput. Vision Pattern Recognition (CVPR)*, 2019. 5

[9] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1271–1278. IEEE, 2016. 3

[10] Kyota Higa and Kota Iwamoto. Robust shelf monitoring using supervised learning for improving on-shelf availability in retail stores. *Sensors*, 19(12), 2019. 1

[11] Swagat Kumar, Geetika Sharma, Nishant Kejriwal, Saumil Jain, Madhvi Kamra, Brijendra Singh, and Vishal Kumar Chauhan. Remote retail monitoring and stock assessment using mobile robots. In *2014 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pages 1–6, 2014. 2

[12] Benjamin Lewandowski, Tim Wengefeld, Sabine Müller, Mathias Jenny, Sebastian Glende, Christof Schröter, Andreas Bley, and Horst-Michael Gross. Socially compliant human-robot interaction for autonomous scanning tasks in supermarket environments. In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 363–370, 2020. 2

[13] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. Yolov6: A single-stage object detection framework for industrial applications, 2022. 5

[14] Annalisa Milella, Antonio Petitti, Roberto Marani, Grazia Cicirelli, and Tiziana D'orazio. Towards intelligent retail: Automated on-shelf availability estimation using a depth camera. *IEEE Access*, 8:19353–19363, 2020. 1

[15] Marina Paolanti, Mirco Sturari, Adriano Mancini, Primo Zingaretti, and Emanuele Frontoni. Mobile robot for retail surveying and inventory using visual and textual analysis of monocular pictures based on deep learning. In *2017 European Conference on Mobile Robots (ECMR)*, pages 1–6, 2017. 2

[16] Christoph Roesmann, Wendelin Feiten, Thomas Woesch, Frank Hoffmann, and Torsten Bertram. Trajectory modification considering dynamic constraints of autonomous robots. In *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6, 2012. 4

[17] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 4

[18] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. 4

[19] Ramiz Yilmazer and Derya Birant. Shelf auditing based on image classification using semi-supervised deep learning to increase on-shelf availability in grocery stores. *Sensors*, 21(2), 2021. 1

[20] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2017. 5