Evolving an innovative Vision for Autonomous Robots course*

Arnoud Visser, Joey van der Kaaij, and Shaodi You

University of Amsterdam, The Netherlands

Abstract. Robotics is a fast-developing field, so it is important to keep robotics courses up to date. Without a good understanding of both the foundations and the latest developments, our students are not ready for the current scientific & societal challenges. With a new robot, we finalized the design of a new master's-level robotics course. Our goal was to challenge students to demonstrate visual SLAM with this new robot by the end of the course, and to use the generated 3D-map to navigate a maze. That was an ambitious goal, yet the students reached that level. This course design had three assignments. The complexity of each task increased with each assignment, but gradually enough to allow the students to build on th previous work. The assignments had the right balance between guidance and being open problems, which could be an inspiration for other institutes that like to provide a state-of-the-art robotics vision course.

Keywords: ROS2-based · Navigation · Localization · Mapping · SLAM.

1 Introduction

Robotics is a topic that is included in the curriculum of many different schools and universities. The focus could be on the electronics, mechanics, perception, or control aspect of the robot. Many textbooks try to cover all those aspects [10,16], although when multiple courses are given at the same university, it could be better to concentrate on one aspect per course.

Fig. 1. The RAE robot, build by Luxonis & the UGV Rover, build by WaveShare.

At the University of Amsterdam, one general robotics course is already given in the bachelor's degree in Artificial Intelligence, based on the textbook 'Introduction to Autonomous Mobile Robots' [17], which is a popular choice [10]. This

* This paper is submitted to the AI Robo conference in Nancy.
It will be presented Wednesday September 23, 2026.

course is accompanied by practical sessions based on a simulator and datasets. For the master Artificial Intelligence, a new course 'Vision for Autonomous Robots' is designed [19] based on a real mobile robot platform like the RAE and UGV-Rover (see Fig. 1). In the design we focused on the perceptual aspect, because this is the basis of all autonomous decisions, and apply the covered algorithms directly on the UGV Rover.

As a research university our mission is to combine theory with experimental research skills. This paper first describes for the theory our selection of textbooks, where we focused on textbooks that had the focus on robotic vision. This is followed with our design of practical assignments, followed with a discussion.

2 Textbooks

An academic textbook with a clear focus on Visual SLAM was published in 2021 [12]. This book shows the foundations of many visual algorithms, with examples in C++ using the OpenCV library [1]. The book contains no Python or ROS examples, which would require additional time to apply the code to the UGV Rover. In 2023 & 2024 two new editions of well-known robotics textbooks were published: 'Robotics, Vision and Control - Fundamental Algorithms in Python' [5] and 'Computational Principles of Mobile Robotics' [9]. The 'Robotics, Vision and Control' 2023 edition is accompanied by many Python examples of visual algorithms. The 'Computational Principles of Mobile Robotics' is also Python based and includes several ROS examples, although more focus on navigation, and less on perception algorithms. Not a textbook, but a good overview of the state-of-the-art, is the upcoming SLAM handbook [3]. We have chosen for the textbooks that are accompanied with code.

2.1 Robotics, Vision and Control

The 'Robotics, Vision and Control' is a textbook with a long history [6], which started with creating a robotics toolbox [4]. In 2005, this toolbox was accompanied by a vision toolbox [7]. Those two toolboxes combined led to the 1st edition of the 'Robotics, Vision and Control' textbook.

The 'Robotics, Vision and Control' textbook is quite unique in the sense that it gives a more than solid theoretical background on algorithms, accompanied by the Python code that created the illustrations and implemented the equations. The vision and control aspects are well integrated, in part because both need spatial math (the third toolbox provided by Peter Corke[1]) and involve the description of combined position & orientation in 2D & 3D. The 2nd chapter of the 'Robotics, Vision and Control' textbook directly dives into subjects like transforms, twists and dual quaternions. The 3rd chapter goes deeper into rigid-body transformation with speed and acceleration. Still, it is clear that the book is written by a robotics engineer, because at the end of chapter 3 the transformations are illustrated with the inner working of Inertial Navigation Units.

[1] Spatial Maths for Python

As textbook 'Robotics, Vision and Control' was very useful for the theory, for the practical sessions a little bit less, because of the design decision to stay away from the Robotic Operation System ROS [6], which is the program interface of most current robots [14], including the UGV Rover. ROS is known for its complexity and is infamous for its steep learning-curve, so there is some rationale in this decision. Yet, it is also a missed chance, because one of the general concepts underlying ROS is the coordinate transformation from the transform library [11], which nicely match with the rigid-body transformations that the textbook starts with.

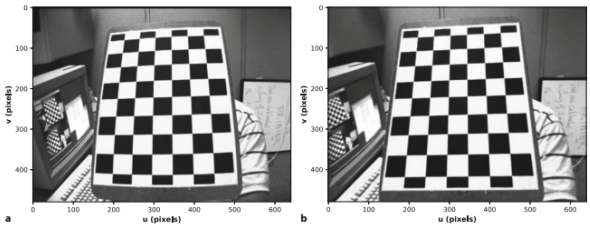

Fig. 2. An example of lens distortion. Courtesy OpenCV / Peter Corke.

For our course, chapters 13 and 14 of the textbook were most useful, where the topics Image Formation and Multiple Views are covered. An example is the description of lens distortion (see Fig. 2.a), including the calibration procedure to undistort the image (see Fig. 2.b)

2.2 Computational Principles of Mobile Robotics

The textbook 'Computational Principles of Mobile Robotics' also has a long history, with a first edition published in 2000, based on a tutorial given in 1993. The third edition is quite recent [9], with for instance a new chapter on deep learning. Yet, for the practical sessions of our course, the most important aspect is the accompanying code[2], which is ROS-based. Unfortunately, the provided code was still in transition from ROS1 to ROS2 (some exercises based on ROS1 Noetic, other exercises based on ROS2 Foxy) which made the ROS learning curve even more challenging.

Chapter 5 of Jenkin's textbook covers visual sensors and their algorithms. The section on feature detectors covers classic edge detectors (see Fig. 3), which we actually used for our first assignment. Chapter 6 demonstrated how convolutional neural networks could be used to follow a line in the simulation. All exercises from this textbook were based on a simulated robot, while our ambition for the 'Vision for Autonomous Robots' course to do the practical sessions with a real robot. However, our first assignment was inspired by Jenkin, to demonstrate that a real UGV Rover could follow a line in the ceiling, as suggested in Chapter 5.

[2] Code examples and exercises

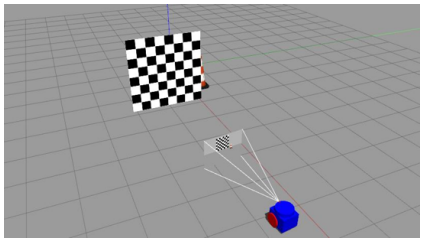

Fig. 3. An example of a simulated robot looking at a chessboard pattern. Courtesy Michael R. Jenkin.

3 Supporting Textbooks

In addition to the two previous textbooks, with a good combination of both Robotics & Computer Vision, we also use two textbooks as supporting material. The first one is clearly focused on Robotics only, the other on Computer Vision only.

3.1 A Concise Introduction to Robot Programming with ROS2

Although Jenkin's textbook comes with supporting material which contains a short introduction to ROS in general, this introduction is rather short. The introduction to ROS2 from Francisco Martín Rico [15] is much more in depth. The textbook concentrates on controlling a robot; both low-level and high-level. This is nicely built around a single *Bump and Go* problem. Initially, this is handled by monitoring just a single ray of the robot's LiDAR, combined with a stop & turn behavior (Chapter 3). This simple control scheme is extended with more complex Virtual Force Fields (Chapter 5). At the end of this chapter the shift is made from the LiDAR measurements to the camera of the robot. The simulated TIAGo robot has a camera in its head, which allows to track obstacles by controlling the pan-tilt in the neck.

Although a camera is used to detect the obstacle, this is done on the color of the object in HSV-color space (see Fig. 4). More advanced computer-vision algorithms are not covered. In chapter 6 this is all combined in Behavior Tree, which allows to combine multiple modules both sequentially and as alternative. This is combined with ROS2's Nav2 package, which allows to do planning and execution of trajectories on a map, without further going into how such a map could be made (with the SLAM toolbox of ROS2).

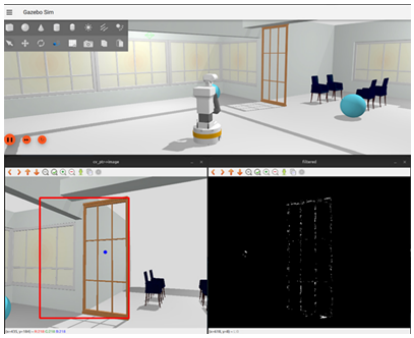

Fig. 4. An example of a simulated TIAGo robot detecting an obstacle. Courtesy Francisco Martín Rico

3.2 Computer Vision Algorithms and Applications

How to detect objects is covered in much more detail in our Computer Vision courses which are based on Szeliski's textbook [18]. The aspects not covered in those courses, partly because they are specific to robotics, are the topics such as Motion Estimation, Structure from Motion and Visual SLAM. In Computer Vision one assumes that the camera follows an unpredictable path, while in robotics one has full control of the viewpoint. In addition, in computer vision an estimation of disparity is often enough, while in robotics metric information is essential to be able to navigate.

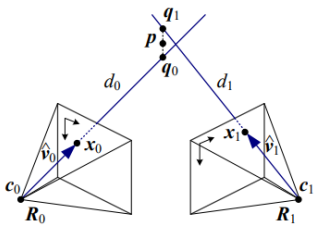

Fig. 5. 3D point triangulation by finding the point $\mathbf{p}$ that lies nearest to all of the optical rays $\mathbf{c}_j + d_j\hat{\mathbf{v}}_j$. Courtesy Szeliski.

For this course, we have concentrated on Chapter 9 and 11 of Szeliski's textbook [18]. For instance, the formulation of 3D point triangulation (see Fig. 5) is very useful for the 2^{nd} assignment. Note that this same topic is also extensively covered in Chapter 14 of Peter Corke's textbook [5].

4 Lectures

In the lecture series, a range of topics were covered. Although for each lecture the corresponding sections in the two textbooks [5,9] were indicated, the lectures themselves were a combination of slides from the courses Autonomous Mobile Robots & Vision Algorithms for Mobile Robotics, both from the ETH Zürich. Especially the topics covered in Vision Algorithms for Mobile Robotics were in line with our approach, concentrating on the geometrical aspects of vision.

Topics covered included:

- Perspective camera
- Camera calibration
- Lens distortion
- Edge detection
- Camera localization
- Kalman Filters
- Triangulation
- Epipolar geometry
- Structure from Motion
- Visual Odometry
- Visual SLAM
- Path Planning

Note that Peter Corke's textbook [5] also has a strong focus on the geometrical aspects of vision & robot control.

5 The UGV Rover

Most commercial robots are now ROS based [14], so to allow the students to build up experience with such a ROS-based system, we selected the UGV Rover. This was based on the following features:

- The UGV Rover has both RGB, a RGB-D and 2D LiDAR sensor, which would allow to compare and combine those sensor modalities when building a 2D or 3D visual map of the environment
- The UGV Rover has a pan-tilt unit, which allows to control the view-point, which allows to demonstrate active vision
- The UGV Rover has a NVIDIA Jetson Orin on-board, which allows to run multiple complex ROS2 nodes at the same time, where the computer vision modules can make use of the NVIDIA GPU.

The Nvidia Jetson Orin delivers up to 67 TOPS of AI performance, which makes it possible to run Vision Transformers such as DINOv2 on-board. The Jetson Orin basis is Nvidia JetPack 6.0, which has a Ubuntu Linux 22.04.05 kernel, including dedicated Nvidia drivers and the full CUDA 12.6 toolbox.

The UGV Rover drivers are implemented as ROS nodes. The idea of ROS drivers is that a number of nodes are launched, each publishing one or more topics. For the UGV Rover such topics can be the pan-tilt camera images, or the current voltage level. Other ROS drivers are intended to control the robot, not publishing but listening to topics. For the UGV Rover robot this is for instance the `cmd_vel` and `led_ctrl`, which respectively drives the robot around and turn the head-lights on/off.

6 Assignments

For the 8-week course, we created 3 assignments. The first assignment had to be finished in the 2nd week, the second assignment in the 5th week, the third assignment was reduced in complexity so that it could be finished in the 7th week. The grades of the second and third assignments were weighted twice as much as the grade for the first assignment.

The first assignment was designed to be a warm-up assignment, which allowed the students to become familiar with ROS and the UGV Rover and its sensor suite. This process required more time and effort from our students, even with the tutorials provided. However, the experience gained during the first two weeks was very valuable in the following weeks.

6.1 Assignment I: Line Following

Line Following is a classic task for a robot, although in most cases one has to follow a line on the ground; here we choose to follow the ceiling lights.

The ceiling lights are bright objects, which allows them to be recognized as bright straight lines (see Fig. 6). Note that not only the four ceiling lights are visible, but also the reflection on the left wall and bright light from the outside window at the right.

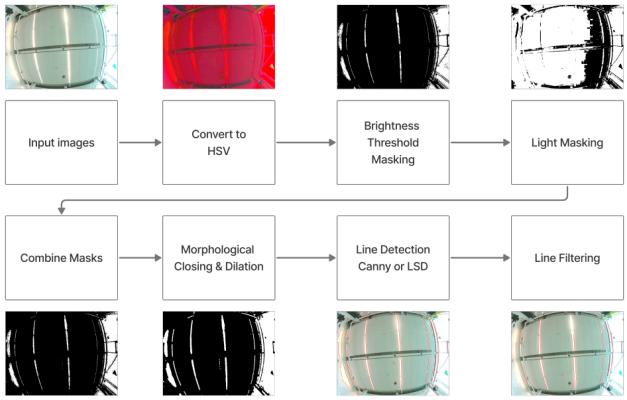

Fig. 6. Combining Canny edge with a Hough transform into a Line Segment Detection algorithm (LSD). Courtesy Silvério *et al.*

After that, the grayscale image is thresholded so that only high-luminance pixels remain, those pixels can be combined into the lines with the classic Canny edge detection [2] or LSD [20] algorithm. Although the images are preprocessed, not all ceiling lights are connected, while other regions of interest are still visible. However, all ceiling lights have an equivalent orientation, which can be analyzed with the Hough transform [8].

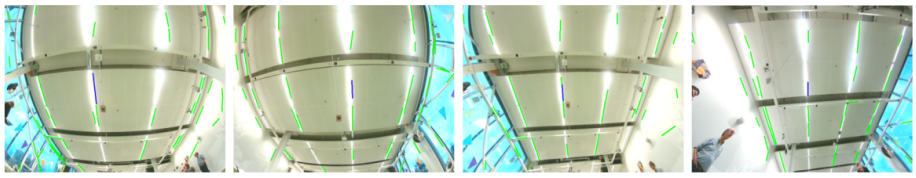

Fig. 7. Two examples of unrectified and two examples of rectified images. Courtesy Silvério *et al.*

The selection of straight lines with the Hough transform performs better when the lines are perfectly straight. In Fig. 6 one could still see the barrel distortion of the top camera pointing to the ceiling. The calibration parameters provided in the `camera_info`-topic of ROS could be used to rectify the image, although one could see in Fig. 7 even in the rectified images still some distortion in the upper-left corner. This no further disturbs the process; the ceiling lines clearly stand out as long lines with a consistent orientation. The overall orientation of the room, which is determined from those ceiling lines and equivalent with the direction the robot should move, is indicated with a small blue line in Fig. 7

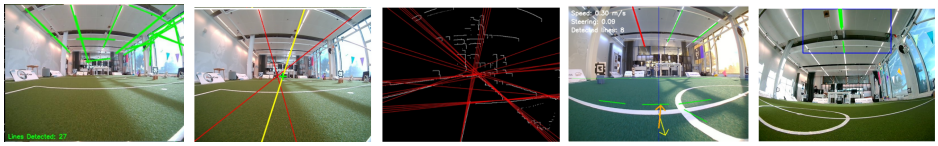

Fig. 8. Using the vanishing point as target to drive the UGV Rover to. Courtesy Buijs, Petrov, Bibo, Guerena Gonzalez, Sverdlov *et al.*

The detection of the ceiling lines could also be done when the pan-tilt camera was directed to the front of the UGV rover. In that case the perspective distortion showing the ceiling light lines all oriented towards a vanishing point (see Fig. 8). The students were also able to detect the ceiling lines for this configuration, although the white lines on the field provided additional distractions. Also note in Fig. 8 the diversity of solutions presented by the students; after starting with Canny edge to detect lines, every student team took another filtering approach.

6.2 Assignment II: Detection & Localization

Localization is estimating the current position of the robot based on a provided map. For this assignment, the provided map consisted of a set of AprilTag markers from the 41h12 tag family, which were placed on objects around the field. The ground-truth position of each of these fiducial makers was measured with a digital laser rangefinder (Bosch Zamo 3). In total 9 of those markers were in-

stalled along the field with a large (inner) size of 16cm × 16cm, to allow them to be recognized at distances of 2m or more.

Fig. 9. Examples of marker detection around the field. In both images one tag is detected (blue square), while one marker to the left is not detected. Courtesy Dragomir *et al.*

In principle the AprilTag markers are easy to detect, because there are several libraries available specialized in detecting those markers. Yet, the students were asked to analyze the failure cases. Especially markers placed far away and at an angle were not always detected.

The students came with several measures to improve the detection rate, such as remove quad decimation, to use adaptive histogram equalization to reduce the effect of backlighting and overexposure, Upscaling the images with bilinear interpolation, multiscale detection and to perform temporal smoothing. With such measures, one of the groups nearly doubled the detection rate from 17 detected markers to 30 detected markers as visible in 37 images, even under difficult lighting conditions (see Fig. 10). Note that the students worked enthusiastically until late at night at their assignments.

Fig. 10. Examples of successful marker detection at large distances and difficult lighting conditions. Courtesy Guerena Gonzalez *et al.*

The size of the fiducial markers was known, so both the angle and distance to the markers could be estimated. The students first measured the error in this estimation. The detection procedure could give a correction to the estimated distance as large as 25% of the actual distance (see Fig. 11). Subsequently, the distance estimate was used to perform triangulation to compute the location from where the observation of multiple fiducial markers was made.

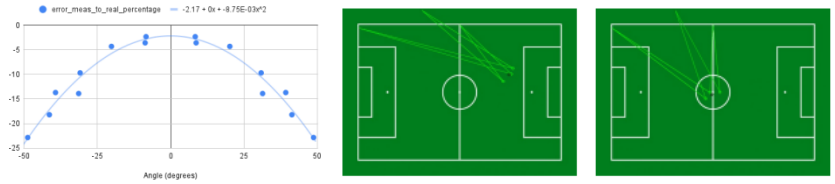

Fig. 11. Measurements of the estimated distance, depending on the angle. The difference between a position estimated based on a uncorrected and corrected distance can be quite large. Courtesy Guerena Bibo *et al.*

The goal of the assignment was to move as accurately as possible from a location in the center of the field to a location in the penalty area. While moving, the robot received observations of the markers (sometimes none are visible, sometimes two or three). Those observations were combined with the movement commands in an Extended Kalman filter to obtain a real-time position.

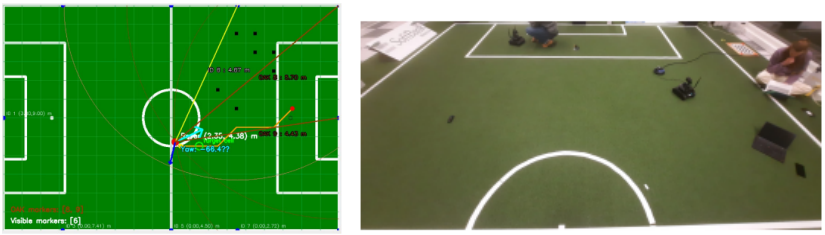

Fig. 12. A plan to the final location (a point inside the penalty area, indicated by a duckie) and a situation with several black obstacles on the field. Courtesy Guerena Gonzalez *et al* and Petrov *et al.*

In most cases, the UGV rovers could navigate accurately to the target location (see Fig. 12). However, not every behavior was very robust against disturbances. A few wrong observations could set the robot astray, which was difficult to recover because at the side of the field less markers are visible. An Extended Kalman Filter is a good tracking algorithm, but has to be reinitialized once the track is lost.

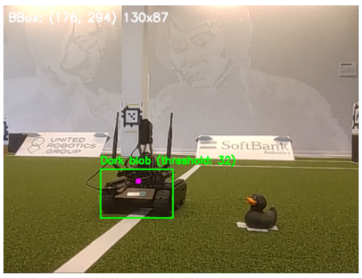

Fig. 13. Detecting the obstacle based on color. Courtesy Sverdlov *et al.*

The last part of the assignments was not only to plan for an undisturbed route, but also to avoid obstacles in the form of other UGV rovers that made the attempt before. Detection of other UGV rovers could be done on their color (see Fig. 13), shape or the LiDAR measurement.

6.3 Assignment III: Mapping & Route Planning

The last assignment was to navigate a maze, while building a visual map of the maze. Like the previous assignment a drive through the mini-version of the maze was pre-recorded as a rosbag, which allowed the students to experiment with Visual SLAM algorithms directly. Based on that experience, the student teams could record their own rosbag driving through the maze (see Fig. 14).

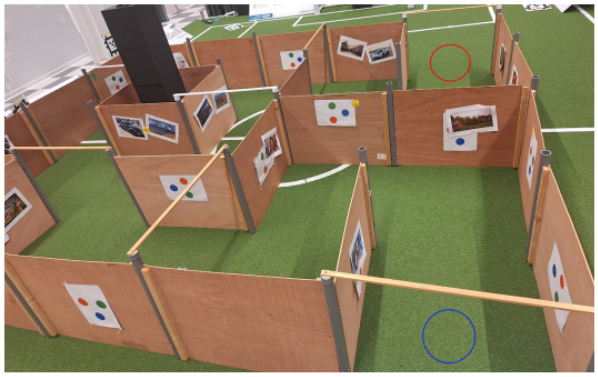

Fig. 14. The maze, with the walls covered with random images to provide enough distinctive texture. The entrance and exit are indicated in respectively a blue and a red circle. Courtesy de Bibo *et al.*

The students could start navigating based on the 2D occupancy map provided by the LiDAR, which was integrated with the RTAB-Map module installed on

the UGV Rover (see Fig. 15). The UGV rover is also equipped with the OAK-D sensor, which could provide colored point-clouds, although such a 3D map has too much noise to be directly usable for navigation.

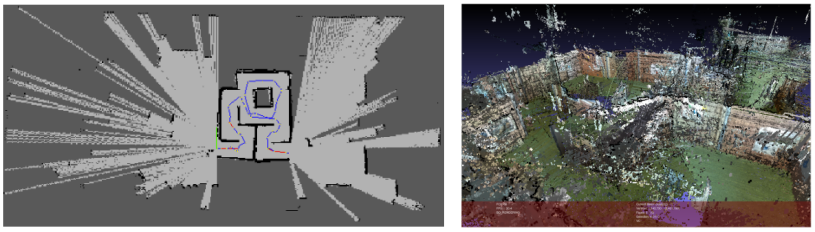

Fig. 15. A 2D occupancy map based on the LiDAR measurements and a 3D point cloud based on the RGBD measurements of the OAK-D light. Courtesy Bibo *et al.*

Part of the assignment was that the students made a 3D map based on pure visual information, using the pan-tilt camera on top of the robot. The RTAB-Map module is already capable of generating pure 3D visual maps, although more modern algorithms like VGGT [21] could provide more accurate 3D maps (see Fig 16). Note that the VGGT map is so accurate that from above it looks like there is an additional slightly lighter colored wall at both the entrance and exit, while in reality it is the small bar holding the walls stable (see Fig 14).

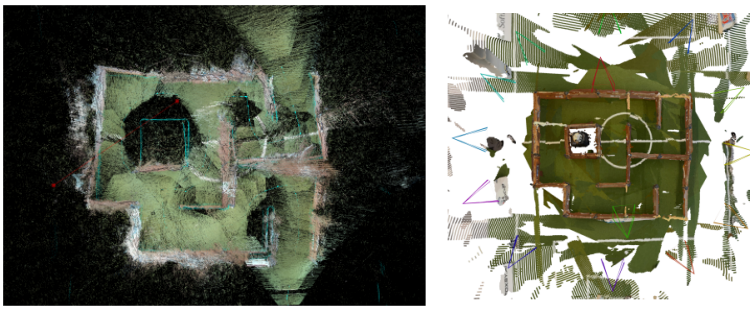

Fig. 16. A 3D visual map created by RTAB-Map and VGGT. Courtesy Brouwer *et al.*

The 3D reconstruction could be projected onto a 2D grid. The students experimented with the pipeline to project the 3D point-cloud on the ground plane, perform ray-casting from the robot poses to obtain an occupancy grid, which the NAV2 module [13] can convert to a costmap which can be used for navigation (see Fig. 17).

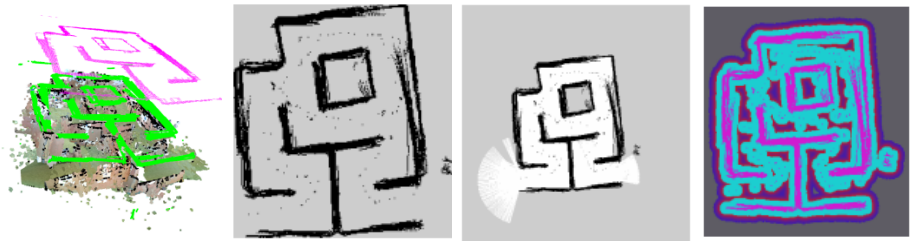

Fig. 17. Conversion of 3D visual map to a 2D costmap. Courtesy Brouwer & Sverdlov *et al.*

Once a representation of the 2D free space is available (including enough margin to stay clear of the walls), path planning can be performed. Before this can be done, one problem has to be solved. The actual 3D maps cover the whole lab, and going around the maze is always a faster route than going through the maze (see Fig. 18). This can be solved by creating artificial barriers (Fig. 18) or by limiting the range of the rays while creating the occupancy map (Fig. 17).

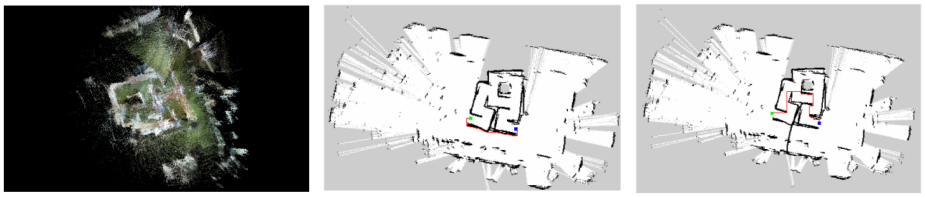

Fig. 18. The free space around the maze. Note the shortest route between the entrance and exit around the outer wall in the middle figure. Courtesy Silvério *et al.*

Once the focus of the path planning is on the inner part of the maze, algorithms like the classic Dijkstra and A* can be applied (see Fig 19). Note that the same path is found by both algorithms; A* was only more efficient in finding the path.

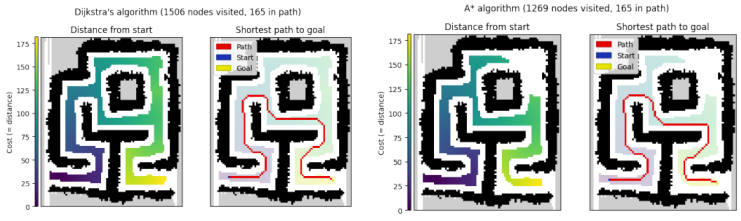

Fig. 19. Path-planning in the free space of the maze. Courtesy Bibo *et al.*

With such a planned path, based on the Nav2 cost-map, the students could navigate the full maze (see Fig. 20). The navigation was based on a pre-processed map, it was not the intention to generate a 3D map, convert it to 2D costmap and perform the path-planning all in real-time. It was more important that the students experienced all aspects of visual navigation.

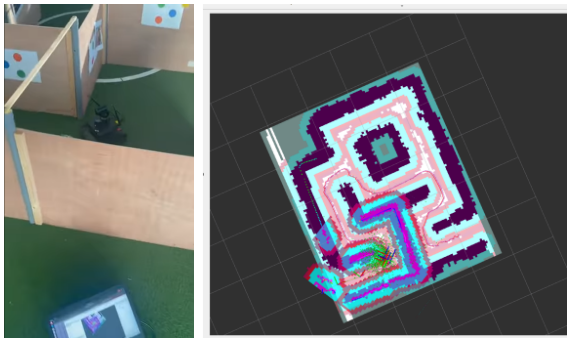

Fig. 20. A drive through the maze. Courtesy Bibo *et al.*

7 Discussion

Several lessons can be learned from this design of a 'Vision for Autonomous Robots'. First, the theoretical side. Robotics has many aspects which could be covered in a course. Perception, and especially vision, for robotics is a rich subtopic enough to be covered by a textbook on its own. Peter Cork's textbook 'Robots, Vision and Control' alone is 824 pages.

For this course two textbooks are chosen that are accompanied with code. This makes the theory concrete, with clear examples how the theory can be applied. Still, in the practical assignments the theory has to be applied slightly different, so the students have to rewrite the code for a new application. No further guidance is given how to do that in the textbooks, that is a skill they have to learn themselves (supported by the teaching assistants).

The three assignments have a nice build up of complexity. The first assignment lets the students get acquainted with the perception-act loop. The stearing actions are still reactive, based on the orientation of the ceiling lights the robot perceives at that moment. In the second assignment the students have combine the current measurements with the measurements done in the past, to be able to localize. This can still be done with a Kalman filter, which keeps track of current estimates and corresponding uncertainties. In the last assignments all past measurements are combined in a Visual 3D map. In the last assignment the power of ROS also becomes really clear; so many modules have to be combined, partly developed by the students themselves, but many other from the extensive ROS libraries. At that moment it becomes more an integration and configuration task.

Performing these experiments on a real robot learns the students to take care of the circumstances; reality is quite different from a fixed dataset or a simulation. Already the first assignment, following lines in the ceiling, was much harder than the students expected. They detected not only the ceiling lights, but many more lines, and had to filter to select the correct ones. Providing datasets in the form of rosbags helped to get the students started with the assignments. What also supported the students that each assignment consisted of ± 10 concrete steps, which guided them through the problem. Each of the steps was formulated in such way that the students had the freedom to come with their own solutions.

8 Conclusion

At the University of Amsterdam, a robotics course was designed, with a clear focus on the perception side. In the first year the course was given [19] the master students were already able to integrate the concepts covered in two recent robotics textbooks [5,9] directly on a robot. After two weeks, the students switched without problems between ROS topics and were able to feed the sensor streams into advanced OpenCV algorithms. So, a lesson learned for every lecturer who is preparing a course at this level: getting familiar with ROS does not conflict with going in depth with concepts of vision and control.

The major problem they encountered was the time needed to experiment with the robot. The original RAE robot quickly depleted its battery, with limited the time to experiment. The new UGV Rover robot was equipped with a better battery (the UGV Rover) and could do part of the processing on-board. To support the students, we also provided ros-bags at the start of each assignment, which allowed the students to have a quick start. Halfway they recorded their own rosbags, once they had their own ideas for the experiment. The students loved the experience, worked very hard on the assignments, and learned a lot. For the next editions the challenge is to further reduce the workload, by providing more support in the form of tutorials on the different aspects that were covered in the assignments.

With this new platform, we like to work on the same assignments, which were perceived as interesting, challenging, and fun by the students. With those three assignments, topics like stereo vision, visual odometry and visual SLAM can be covered, which was the intention of designing this course.

Acknowledgement

We would like to thank all students who participated in the second edition of this course for their contribution. In this paper figures and highlights from their project reports were used: the illustrations are from Luc Buijs, Meher Changlani, Rénan van Dijk & Adrian Ionescu; Mara-Iuliana Dragomir & Rick Haakman; Nikola Petrov, Akshay Sardjoe Missier, Weronika Orska & Viktória Pravdová; Thomas Brouwer, Rick van der Veen & Bart de Zwart; Julian Bibo, Madelon Bernardy & Julia Blauuboer; Alejandro Guerena Gonzalez, Lukas Bierling, Oliver van Erven & Quinten van Engelen; Alessio Silvério, Aïda Asma & Fiona Nagelhout; Marom Sverdlov, Wojciech Trejter, Koen Veldhorst & Maxim Voronin.

References

1. Bradski, G.: The opencv library. Dr. Dobb's Journal of Software Tools **25**(11), 120, 122–125 (Nov 2000)
2. Canny, J.: A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence **8**(6), 679–698 (Nov 1986)
3. Carlone, L., Kima, A., Barfoot, T., Cremers, D., Dellaert, F. (eds.): SLAM Handbook - From Localization and Mapping to Spatial Intelligence. Cambridge University Press, public release, compiled on April 21, 2026
4. Corke, P.: A robotics toolbox for matlab. IEEE Robotics & Automation Magazine **3**(1), 24–32 (Mar 1996)
5. Corke, P.: Robotics, Vision and Control: fundamental algorithms in Python, Springer Tracts in Advanced Robotics, vol. 146. Springer (May 2023)
6. Corke, P., Haviland, J.: Not your grandmother's toolbox – the robotics toolbox reinvented for python. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 11357–11363 (Oct 2021)
7. Corke, P.: The machine vision toolbox: a matlab toolbox for vision and vision-based control. IEEE Robotics & Automation Magazine **12**(4), 16–25 (Dec 2005)
8. Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. Commun. ACM **15**(1), 11–15 (Jan 1972)
9. Dudek, G., Jenkin, M.: Computational principles of mobile robotics. Cambridge university press (Feb 2024)
10. Esposito, J.M.: The state of robotics education: Proposed goals for positively transforming robotics education at postsecondary institutions. IEEE Robotics & Automation Magazine **24**(3), 157–164 (Jul 2017)
11. Foote, T.: tf: The transform library. In: 2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA). pp. 1–6. IEEE (Jul 2013)
12. Gao, X., Zhang, T.: Introduction to visual SLAM: from theory to practice. Springer Nature (Sep 2021)
13. Macenski, S., Martín, F., White, R., Clavero, J.G.: The marathon 2: A navigation system. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 2718–2725. IEEE (Oct 2020)
14. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y., et al.: Ros: an open-source robot operating system. In: ICRA workshop on open source software (May 2009)
15. Rico, F.M.: A concise introduction to robot programming with ROS2. Chapman and Hall/CRC (Sep 2022)
16. Siciliano, B. (ed.): Robotics Goes MOOC. Springer Nature (Jun 2005)
17. Siegwart, R., Nourbakhsh, I.R., Scaramuzza, D.: Introduction to autonomous mobile robots. MIT press, 2nd edn. (Feb 2011)
18. Szeliski, R.: Computer Vision Algorithms and Applications. Springer (Jan 2022)
19. Visser, A., van der Kaaij, J., Bi, Q., You, S.: Designing an innovative vision for autonomous robots course. In: Proc. of the 19th International Conference on Intelligent Autonomous Systems (Jul 2025)
20. Von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: Lsd: A fast line segment detector with a false detection control. IEEE transactions on pattern analysis and machine intelligence **32**(4), 722–732 (Dec 2008)
21. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 5294–5306 (Jun 2025)