Digital Twins: A 3D simulation approach to test validation with an autonomous vehicle

Nalia Sánchez [*] **Marcos V. Cruz** [*] **Rafael F. Santos** [*]
Rubén Hernandez [**] **William G. Almeida** [*] **Giovani Bernardes** [*]

[*] *Federal University of Itajuba, Institute of technological Sciences - ICT/UNIFEI, Laboratory of Robotics, intelligent and Complex Systems - RobSic, 35903-087 Itabira-MG*
[**] *Universidad Militar Nueva Granada, Colombia*

Abstract:
Within the Digital Twins context, efforts are being made to minimize costs, increase safety, and speed up tests within a specific application, based on computer graphics tools for three-dimensional simulations. Thus, as a way to mitigate mainly the risks with the safety issue involving activities with autonomous vehicles, the present work proposes the modeling and structuring of an electric golf cart in a 3D virtual environment, so that it can serve for studies in the areas of perception, navigation, and control. Therefore, taking as a reference the proper vehicle existing at the university, Blender software was used together with the Gazebo simulator to perform the validation of this simulation environment. Different open source Environment Mapping (SLAM), Pattern Recognition (YOLO), and Autonomous Navigation algorithms were integrated to minimize possible errors regarding their integration in the actual vehicle. Finally, validation tests were performed on the Yolo algorithm, resulting in an accuracy of 98.9% with a margin of error of 1.1% in the identification of objects.
Resumo:
Dentro do contexto de *Digital Twins*, esforços estão sendo realizados para minimizar os custos, aumentar a segurança e agilizar ensaios dentro de uma aplicação específica, fundamentados em ferramentas de computação gráfica para simulações tridimensionais. Sendo assim, como forma de mitigar principalmente riscos com a questão da segurança envolvendo atividades com veículos autônomos, o presente trabalho propõe a realização da modelagem e estruturação de um carro de golfe elétrico em ambiente virtual 3D, de maneira que este possa servir de estudos nas áreas de percepção, navegação e controle. Portanto, tomando como referência o veículo real existente na universidade, foi utilizado o software Blender juntamente com o simulador GAZEBO para que, não somente o veículo, mais também os principais pontos do ambiente por onde o veículo irá ser ensaiados no cenário real fossem incluídos e representados no ambiente de simulação 3D. Para validação deste ambiente de simulação, diferentes algoritmos de código aberto para Mapeamento de Ambiente (SLAM), Reconhecimento de Padrões (YOLO) e Navegação Autônoma foram integrados afim de minimizar possíveis erros quanto sua integração no veículo real. Por fim, foi realizado testes de validação no algoritmo Yolo, obtendo como resultado uma precisão de 98,9% com uma margem de erro de 1,1% na identificação de objetos.

Keywords: Digital Twins, simulation, 3D modeling, autonomous Navigation, object detection.
Palavras-chaves: Gêmeos Digitais, simulação, modelagem 3D, navegação autônoma, detecção de objetos.

1. INTRODUCTION

The implementation of low-cost technologies in software and hardware has given rise to many new automation and autonomy solutions. Since, their application in the industry increases productivity and improves different processes, for example, IoT, AI, Industry 4.0, or Big Data (Redeker et al. (2021); Faz-Mendoza et al. (2020)); demonstrates additional benefits in the use of resources and improved productivity.

Given this, when undertaking a project, it is necessary to carry out a series of tests, not only to start it up but also to periodically reevaluate it. Consequently, there are simulation tools such as the software CoppeliaSim, Webots, Gazebo, OpenRave, and RodoDK, among others (Jakubiec (2018)), which provide a realistic representation of the physical environment of the virtual environment. Given this, there are great challenges today, regarding the final design, since, if you know how to implement these tools, can reduce time, and costs, optimize the manufacture of the product and even allow the identification of problems and errors that could occur in real life.

These simulation tools are currently being used for the realization of autonomous vehicles. However, this is a topic that requires a lot of research, since the implementation of a well-designed simulator will allow us to quickly test algorithms, design collisions, and even perform tests of different real-world scenarios.

Gazebo is a simulator that provides realistic sensor information and interactions between potentially plausible objects, including an accurate simulation of rigid body physics. This simulator can be integrated with ROS (Robot Operating System), which provides necessary interfaces to develop robots, environments, and development of intelligent algorithms for planning, navigation, and testing of the environment with specific applications in various areas: hospital, industrial, business, etc (Xu et al. (2021); Sai Sahith Velamala (2017); Hussein et al. (2018); Marian et al. (2020); Mario Gluhaković (2020); Rivera et al. (2019); Quang et al. (2019)).

In the literature, different authors have used such tools. Banjanovic-Mehmedovic et al. (2021b) used ROS, Gazebo, and Rviz in an application where the robot navigates in an environment that is responsible for solving problems related to disinfection in a Covid-19 contaminated environment. On the other hand, Ahmed et al. (2019) performed an application in which they perform activities related to industrial maintenance considered high risk and difficult to access, subject to chemicals, hazardous substances, or biological substances, used in an unmanned aerial vehicle. Also, Arango et al. (2020) presents the development process of the ROS-based Drive-By-Wire system, designed for an open-source autonomous electric vehicle prototype, which implements a manual/automatic interchange system, allowing the driver to activate autonomous driving and safely take control of the vehicle at any time. This and other authors (Villa et al. (2020); Tian et al. (2021); Yang and Chi (2021)) have implemented these new technologies utilizing simulations to ensure better control of the project. In addition, allows tasks to be carried out in efficient ways in terms of latency such as energy consumption, among others.

However, when working with autonomous vehicles, this type of driving without human intervention requires continuous processing of the images captured by the vehicle during its journey, to accurately determine the path to follow, as well as the possible presence of obstacles along the way. Given the large amount of data to be processed and the essential response of the vehicle in real-time, it is essential to explore new architectures to perform these tasks efficiently both in terms of latency and energy consumption. For this reason, this work proposes to implement an initial phase that allows simulations describing the interaction of an autonomous vehicle with the study environment, whose models are focused on scenarios of the Federal University of Itabira (UNIFEI), where the autonomous vehicle is being designed and where real experiments will be reproduced. Based on the above, this research contributes to the following results:

- Blender modeling of an environment that includes the structure of the Unifei campus and the vehicle (golf cart) of which it is in the process of integrating sensors that will be used in simulations in a real environment.
- They carried validation of the Yolo algorithm out through the calculation of a confusion matrix, which allows demonstrating the performance of the algorithm with its respective margin of error. Likewise, the integration with the RQT application was performed, which evidences the open-source object detection method and the Ros integration.
- It generated a map from the integration of the SLAM cartographer that provides simultaneous real-time location and mapping on the 2D plane of the University. In addition, they carried integration out “Nav2” package (Stack (2020)) which results in the desired route in the virtual environment, integrating a particle filter and the localization algorithm (AMCL, adaptive localization Monte Carlo).

This article is divided into the construction of the robotic system and the simulation environment, application implementation, results, and conclusions. The system architecture, modeling and environment configuration are shown as a ROS-based solution that can be transferred to different simulation platforms from which they implement autonomous navigation, perception, and 2D mapping.

2. 3D MODELLING AND STRUCTURING OF THE ROBOTIC PLATFORM

2.1 Context of development

The figure 1 presents the background of the development of this work, divided into two stages. The first stage includes the design of the Unifei University and the golf cart, which were designed in Blender software, allowing the modeling, rendering, and creation of simulations, which in this case allows the integration with Gazebo, which allows the second stage. The second stage includes the integration of the model to Gazebo, which allows the transfer of information about the interaction of the vehicle to the physical environment, it also allows the integration of different applications, in this case, three applications are implemented, which serve to validate the design of the car and the environment. From these two stages, they expected to obtain results of the mapping, navigation, visualization, and pattern recognition, which are intended to be implemented in the real golf cart.

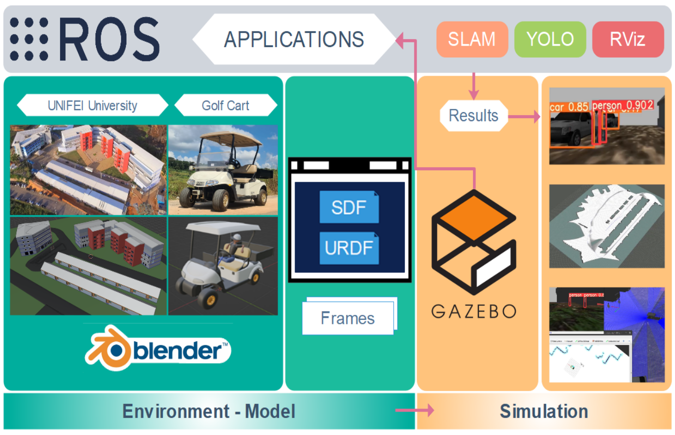


Figure 1. Phases for the implementation of a simulated autonomous vehicle.

The integration through Gazebo uses the URDF and SDF files. These formats are files are formats established to describe the robot structure in the RViz visualization software and the Gazebo physical simulation software. Both files satisfy the simulation needs in a Gazebo environment, resulting in the implementation of SLAM (navigation) and Yolo (sensing). In this way, these applications can be implemented in a real environment from which the simulations generated in the software will be validated, they describe each stage below:

2.2 Environment

To create the simulation environment, we started with two steps. First, modeling the Unifei Itabira campus (Figure 2). Second, modeling the golf cart, mentioned earlier. Both models were modeled in Blender, the university was modeled using real images of the environment. Similarly, the golf cart (model EZGO RXV) was modeled using the images and measurements of the actual vehicle present at the university.

The modeling of the environment and vehicle through the Blender software allows these designs to be saved in different extensions such as .obj, .stl, .fbx, among others. Such files allow the integration of the models (environment and vehicle) in the Gazebo platform that will be used as visualization and collision components.

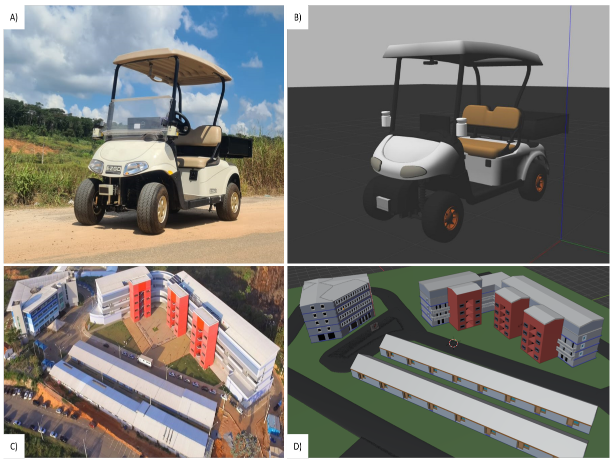


Figure 2. Real environment and physical environment. A) golf cart, B) golf cart modeling, C) Unifei University where the project is being carried out, and D) modeling of the university environment.

2.3 Modeling

The construction of the physical model of a robot starts from the SDF and URDF files. Both files contain information about the fixed and moving parts that make up the robot, as well as the connections between them and the sensing that is defined through the concepts of links, joints, and plugins that correspond to the robot members, joints, and sensor integration, respectively.

The URDF (Unified Robot Description Format) file is obtained from the transformation frames of each member of the vehicle, ie, the position of the base of the vehicle about lasers, camera, IMU, etc. Figure 3 shows the golf cart in the Gazebo environment, where the integration of different sensors was carried out and contains the visual representation of the concepts of the physical model of the vehicle and the transformations between the frames generated in the RViz software.

In addition, once the SDF and URDF files are generated, both are executed in the Gazebo and Rviz environments. In Figure 3, the vehicle is in the Gazebo environment with all the sensing specifications that were defined in the real vehicle. As shown the vehicle has two 1-layer lasers (located in the front region of the vehicle) and two 4-layer lasers (located in the front region near the headlight and the rear region under the body of the vehicle), the two 1-layer lasers perform a scanning whose angle involves an aperture from 0 to 270°. And the other two 4-layer lasers scan from an angle of 50° to -60°. The other sensors are for orientation (IMU and GPS), stereo camera, and encoders.

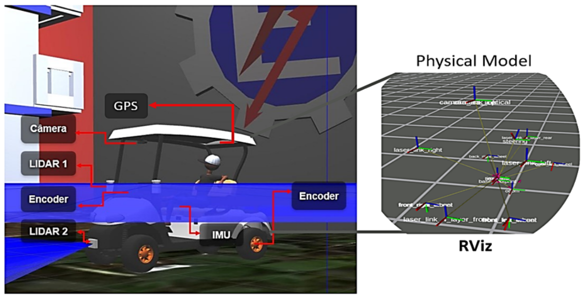


Figure 3. Integration of sensors and visual part of the physical model of the vehicle.

Another way to represent and analyze the transformation frame and the relationship between the vehicle members is through the transformation tree link[1] which has a graphical representation whose aspect is a tree of nodes (frames) that obeys a hierarchy of which establishes the relationship between each part of the vehicle, for example, the relationship between the base of the vehicle (base_link), camera (camera-link) and sensors (imu link, camera link).

2.4 Simulation (environment configuration)

The simulations involve applications focused on mapping, navigation, and pattern recognition. The simultaneous localization and mapping (SLAM) in the simulation environment is done using the vehicle's lasers and odometry sensors. As the vehicle moves through the environment and detects its surroundings, the region detected in its surroundings is recorded and this information contributes to the navigation of the vehicle in the environment. The same is true for pattern recognition (YOLO) via the stereo camera.

3. DEPLOYED APPLICATIONS

When obtaining the simulation model, different applications that can be implemented in this system were found, which are the following:

[1] https://drive.google.com/file/d/1pjIkLkda203lOi2utAYFF5zzxFxgQPQo/view

3.1 Simultaneous localization and mapping (SLAM)

To create the map of the environment, a package called SLAM cartographer developed by Google was used. The generation of the 2D map is based on the data generated by lidar sensors, odometry, and IMU, whose sensors are necessary for the implementation of this package, therefore first the vehicle was modeled and then this algorithm was used. SLAM refers to the problem of constructing a map of an unknown environment by a mobile robot, while allowing the use of the map for localization and navigation in the environment (Banjanovic-Mehmedovic et al. (2021a)). The main function of SLAM is to analyze the input data to determine the position of the vehicle and build a map of the environment where it can move autonomously.

SLAM, on the other hand, allows the marking of trajectories, which can be developed from different filtering or smoothing methods, such as Kalman filters (EKF, UKF) and particle filters (Xuexi et al. (2019); Valencia et al. (2011a,b); Burger et al. (2019)). In this case, they divided the implementation of the mapping package into two types: local SLAM and global SLAM. The local SLAM consists of the construction of a time-varying sequence of local sub-maps, and the global SLAM allows for finding closed-loop constraints, which are demonstrated when a car has to pass a traffic circle, generating a map containing information about the surrounding environment and its obstacles (Figure 5).

3.2 Detection Yolo (You Only Look Once)

For autonomous vehicles, perception is necessary since it is responsible for analyzing the environment surrounding the vehicle, detecting, and recognizing the objects in it, for the acquisition of information, this is usually implemented through various types of sensors, such as cameras, LIDAR, radar, ultrasonic devices, etc. The camera is a sensor that can provide more detailed information about the vehicle environment in terms of high resolution and texture information. Given this, algorithms were created that allow the detection of obstacles and objects in the environment, allowing to perform better control of the environment information.

Similarly, object detection is an advanced form of image classification in which a neural network predicts objects in an image and points to them in the form of bounding boxes (Bjelonic (2016–2018)), i.e., it can perform generalized detection, recognition, or localization tasks in real-world scenarios. There is an algorithm called Yolo (You Only Look Once) that uses classifiers to perform detection using end-to-end neural networks that make bounding box predictions and class probabilities at the same time, it achieves state-of-the-art results outperforming other real-time objects detection algorithms by a wide margin (Corović et al. (2018); Bjelonic (2016–2018)). From this, the integration of this algorithm with Ros allows performing a simulation of the vehicle, to detect obstacles encountered on the road.

This project uses YoloV5 which is previously trained with the COCO dataset, which will identify objects such as cars, people, and trucks, among others. In addition, a ZED stereoscopic camera is simulated which generates the real-time position of the camera and allows to perform environment detection. Figure 4 shows the process followed for the implementation of this algorithm, which is divided into three parts, which are part of the image processing. In the capture stage, the visual image of the environment is obtained to perform a preprocessing that includes noise reduction techniques and detail enhancement. In this case, the segmentation that divides the image according to the objects to be detected, people and cars, is also implemented. After this, comes the description and detection part, which is the process where the characteristics of the image such as size and shape are obtained to determine the recognition and implement it in the graphic part of Gazebo.

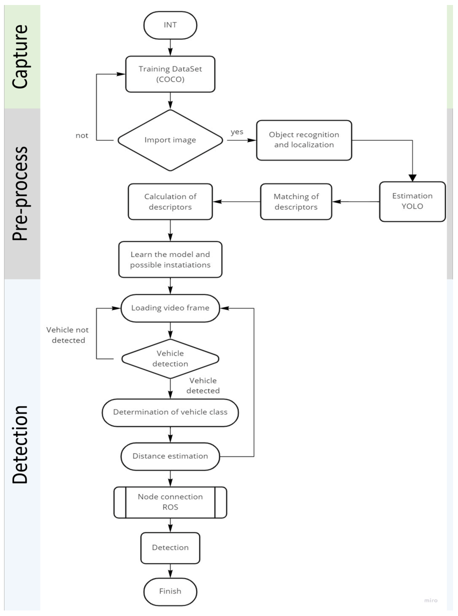


Figure 4. Algorithm implemented for object detection using the Yolo implementation.

4. RESULTS

As a result of the implementation of the simulation, they carried out two validation tests, where the environment model is static, and they generate the robot model in this environment to navigate. They explain these tests below.

4.1 Map and trajectory creation

By navigating the environment, the map shown in Figure 5 was generated. The contour includes the 2D profile of the environment. The white region is the occupancy of the modeled environment (university) in space and the black contours indicate the boundaries of the environment. As the vehicle moves through the university, the map measurements become more accurate through optimization algorithms that assemble sub-maps and cluster into a global map. In addition, during vehicle navigation it was necessary to choose a trajectory (blue curve in Figure 7), speed (from 0 to 5 m/s) and laser data publication rate (30Hz) that could provide a global map with the best assembly of all submaps. Xuexi et al. (2019) demonstrated in practical experiments that map generation is influenced by the velocity of the robot in the environment, the laser data publication rate and the trajectory. The proper choice of the trajectory that provides the best loop is also responsible for the quality and clustering of the submap maps that make up the overall map generation.

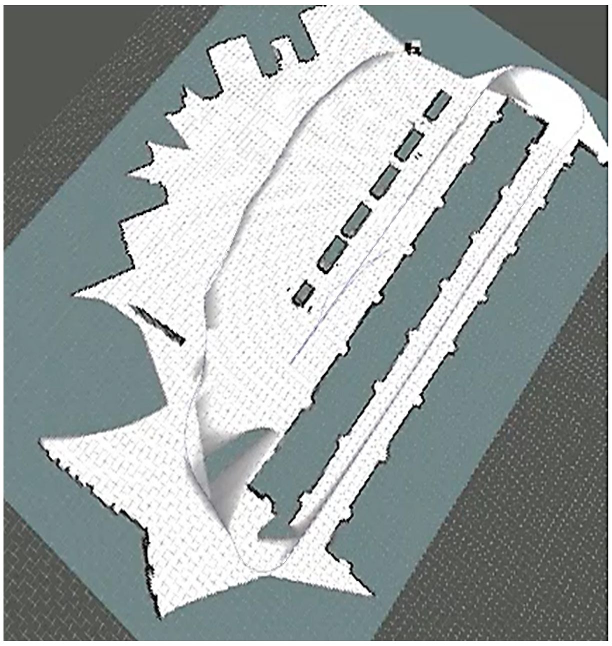


Figure 5. Result of the map obtained according to UNIFEI

The following Figure 6 generated by gui rqt graph shows the result given according to the relationship between the mapping, Gazebo, Rviz, Yolo and teleoperation nodes, as well as other packages responsible for the operation of the SLAM cartographer software.

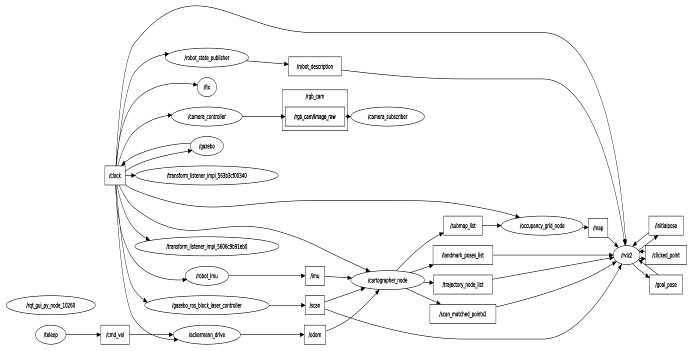


Figure 6. Visualization of the nodes connected to Yolo and Slam.

4.2 Navigation

Once the global map was generated, it was used to make the vehicle navigation in the simulation environment. In this simulation, the Navigation2 software package shows the delimitations of the blue environment that are considered obstacles for vehicle navigation. In addition, the software uses the AMCL (adaptive Monte Carlo) particle filter which has the location of the vehicle on the map and helps the robot to move to the desired point. Figure 7 shows the result of vehicle navigation leaving the map origin and moving to the desired point.

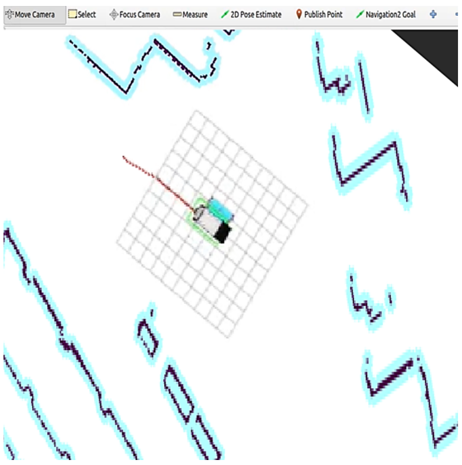


Figure 7. Map showing obstacles and trajectory of the vehicle.

On the other hand, a graph generated by RQT is also obtained as a result, which shows the relationship between the navigation nodes, Gazebo, Rviz, and other packages responsible for the operation of the nav2 software (Operation of the Nav2 software through the node map, link [2]). In the following link [3] the result of the simulation and integration is presented.

4.3 Detection

The image recognition algorithm reports the objects present in the environment, in this case vehicles and people. This information is extremely important for making decisions about vehicle navigation and the safety of the user and people in the environment. The combination of this information, with the map created from the environment, allows better decisions to be made in the autonomous navigation of the vehicle within the University. Figure 8 shows the obtained sensing and the generated map in a teleoperated navigation.

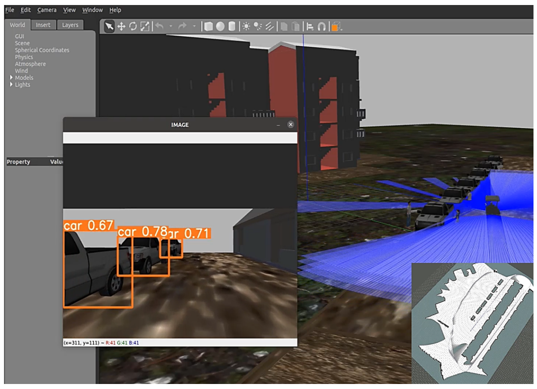


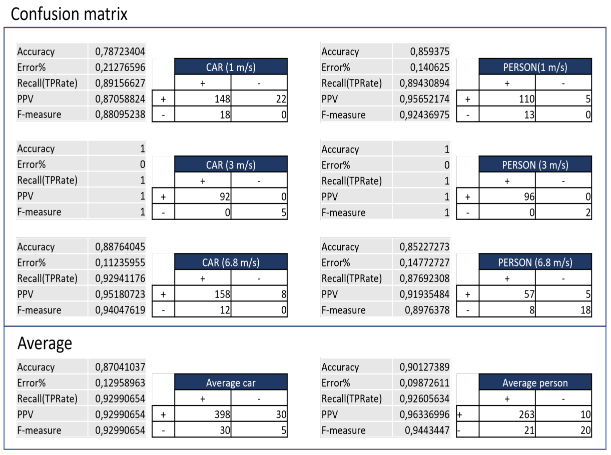
Figure 8. Integration of gazebo and slam for navigation and object detection.

On the other hand, in order to verify that the Yolo algorithm worked correctly, an experiment was carried out consisting of recording a one-minute video, where the car

[2] https://drive.google.com/file/d/1N4mIiThGUH9CZVYMBvj15kdWL4j6PfKj/view?usp=sharing
[3] https://youtu.be/aBmsR5X1hQc

navigates at three different speeds (1, 3 and 6.8 m/s). These videos were analyzed for each second giving the possibility of identifying false positives or false negatives, in order to make a confusion matrix (Khan et al. (2021); Sommer et al. (2020)) that will allow visualizing the performance of the algorithm in terms of object detection.

A confusion matrix was used for each of these objects (cars and people), to obtain precision results that give the proportion of correctly labeled objects to the whole sample. Metrics such as Recall, which is responsible for analyzing how many objects are detected and correctly detected, and the F-measure that determines the harmonic mean of precision and recall at the time of detection are used. The results of the videos are compiled in Figure 9, according to their corresponding precision, error%, recall (TPRate), PPV and F-measure.



Confusion matrix

			CAR (1 m/s)	
Accuracy	0,78723404			
Error%	0,21276596		+	-
Recall(TPRate)	0,89156627			
PPV	0,87058824	+	148	22
F-measure	0,88095238	-	18	0

			PERSON(1 m/s)	
Accuracy	0,859375			
Error%	0,140625		+	-
Recall(TPRate)	0,89430894			
PPV	0,95652174	+	110	5
F-measure	0,92436975	-	13	0

			CAR (3 m/s)	
Accuracy	1			
Error%	0		+	-
Recall(TPRate)	1			
PPV	1	+	92	0
F-measure	1	-	0	5

			PERSON (3 m/s)	
Accuracy	1			
Error%	0		+	-
Recall(TPRate)	1			
PPV	1	+	96	0
F-measure	1	-	0	2

			CAR (6.8 m/s)	
Accuracy	0,88764045			
Error%	0,11235955		+	-
Recall(TPRate)	0,92941176			
PPV	0,95180723	+	158	8
F-measure	0,94047619	-	12	0

			PERSON (6.8 m/s)	
Accuracy	0,85227273			
Error%	0,14772727		+	-
Recall(TPRate)	0,87692308			
PPV	0,91935484	+	57	5
F-measure	0,8976378	-	8	18

Average

			Average car	
Accuracy	0,87041037			
Error%	0,12958963		+	-
Recall(TPRate)	0,92990654			
PPV	0,92990654	+	398	30
F-measure	0,92990654	-	30	5

			Average person	
Accuracy	0,90127389			
Error%	0,09872611		+	-
Recall(TPRate)	0,92605634			
PPV	0,96336996	+	263	10
F-measure	0,9443447	-	21	20

Figure 9. Results of the confusion matrix according to the cars and people detected in three times, obtaining as a result different parameters.

The results of averaging the matrices in Figure 9 show that the YOLO software works since the measured parameters do not reflect a great difference from the results obtained through the average. However, it is clear that, at the time of detection, it has a greater margin of error in cars, since this algorithm works through distance and shape, this means that it identifies cars when it finds something similar to their contour. On the other hand, it is analyzed that the higher the speed, the lower the margin of error. The following link[4] presents an experiment with navigation and object detection.

5. CONCLUSION

The experiments showed that it is possible to perform a simulation of a vehicle at the university (Unifei) through the ROS frameworks and Gazebo software, in addition, it was possible to add some of the sensors that are being implemented in the real golf cart.

As well, they implemented a detection algorithm that allows the use of Yolo, which is capable of detecting objects in different positions, obtaining an accuracy of 98.9% with a margin of error of 1.1% when identifying objects according to their shape, these results were validated with the confusion matrix.

In future work, these applications will be improved and implemented on the vehicle currently in Unifei's facilities. In order to test the simulation, works in the proper environment, which allows you to reduce costs and time.

REFERENCES

Ahmed, A.A., Olumide, A., Akinwa, A., and Chouikha, M. (2019). Constructing 3d maps for dynamic environments using autonomous uavs. In *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, MobiQuitous '19, 504–513. Association for Computing Machinery, New York, NY, USA. doi:10.1145/3360774.3368200. URL `https://doi.org/10.1145/3360774.3368200`.

Arango, J.F., Bergasa, L.M., Revenga, P.A., Barea, R., López-Guillén, E., Gómez-Huélamo, C., Araluce, J., and Gutiérrez, R. (2020). Drive-by-wire development process based on ros for an autonomous electric vehicle. *Sensors*, 20(21). doi:10.3390/s20216121. URL `https://www.mdpi.com/1424-8220/20/21/6121`.

Banjanovic-Mehmedovic, L., Karabegovic, I., Jahic, J., and Omercic, M. (2021a). Optimal path planning of a disinfection mobile robot against covid-19 in a ros-based research platform. *Advances in Production Engineering And Management*, 16, 405–417. doi:10.14743/APEM2021.4.409.

Banjanovic-Mehmedovic, L., Karabegović, I., Jahic, J., and Omercic, M. (2021b). Optimal path planning of a disinfection mobile robot against covid-19 in a ros-based research platform. *Advances in Production Engineering Management*, 16, 405–417. doi:10.14743/apem2021.4.409.

Bjelonic, M. (2016–2018). YOLO ROS: Real-time object detection for ROS. `https://github.com/leggedrobotics/darknet_ros`.

Burger, P., Naujoks, B., and Wuensche, H.J. (2019). Map-aware slam with sparse map features. *IEEE International Conference on Intelligent Robots and Systems*, 347–353. doi:10.1109/IROS40897.2019.8968466.

Corović, A., Ilić, V., Duric, S., Marijan, M., and Pavković, B. (2018). The real-time detection of traffic participants using yolo algorithm. In *2018 26th Telecommunications Forum (TELFOR)*, 1–4. doi:10.1109/TELFOR.2018.8611986.

Faz-Mendoza, A., Gamboa-Rosales, N.K., Medina-Rodríguez, C.E., Casas-Valadez, M.A., Castorena-Robles, A., and López-Robles, J.R. (2020). Intelligent processes in the context of mining 4.0: Trends, research challenges and opportunities. In *2020 International Conference on Decision Aid Sciences and Application (DASA)*, 480–484. doi:10.1109/DASA51403.2020.9317095.

Hussein, A., García, F., and Olaverri-Monreal, C. (2018). Ros and unity based framework for intelligent vehicles control and simulation. In *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, 1–6. doi:10.1109/ICVES.2018.8519522.

Jakubiec, B. (2018). Application of simulation models for programming of robots. *SOCIETY. INTEGRATION. EDUCATION. Proceedings of the International Scientific Conference*, 5, 283. doi:10.17770/sie2018vol1.3214.

[4] https://youtu.be/45qVR0e$_Rwg$

Khan, M.Z., Khan, M.U.G., Saba, T., Razzak, I., Rehman, A., and Bahaj, S.A. (2021). Hot-spot zone detection to tackle covid19 spread by fusing the traditional machine learning and deep learning approaches of computer vision. *IEEE Access*, 9, 100040–100049. doi:10.1109/ACCESS.2021.3094720.

Marian, M., Stîngă, F., Georgescu, M.T., Roibu, H., Popescu, D., and Manta, F. (2020). A ros-based control application for a robotic platform using the gazebo 3d simulator. In *2020 21th International Carpathian Control Conference (ICCC)*, 1–5. doi:10.1109/ICCC49264.2020.9257256.

Mario Gluhaković, M.H. (2020). *2020 Zooming Innovation in Consumer Technologies Conference (ZINC) : Online, 26-27 May 2020.*

Quang, H.D., Manh, T.N., Manh, C.N., Tien, D.P., Van, M.T., Kim, D.H.T., Thanh, V.N.T., and Duan, D.H. (2019). *Mapping and Navigation with Four-wheeled Omnidirectional Mobile Robot Based on Robot Operating System.* IEEE.

Redeker, M., Klarhorst, C., Göllner, D., Quirin, D., Wißbrock, P., Althoff, S., and Hesse, M. (2021). Towards an autonomous application of smart services in industry 4.0. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–4. doi:10.1109/ETFA45728.2021.9613369.

Rivera, Z.B., De Simone, M.C., and Guida, D. (2019). Unmanned ground vehicle modelling in gazebo/ros-based environments. *Machines*, 7(2). doi:10.3390/machines7020042. URL `https://www.mdpi.com/2075-1702/7/2/42`.

Sai Sahith Velamala, Devendra Patil, X.M. (2017). *IEEE ROBIO 2017 : 2017 IEEE International Conference on Robotics and Biomimetics : December 5-8, 2017, Macau SAR, China.*

Sommer, N.M., Velipasalar, S., Hirshfield, L., Lu, Y., and Kakillioglu, B. (2020). Simultaneous and spatiotemporal detection of different levels of activity in multidimensional data. *IEEE Access*, 8, 118205–118218. doi:10.1109/ACCESS.2020.3005633.

Stack, R.N. (2020). Nav2. URL `https://navigation.ros.org/about/index.html#about`.

Tian, S., Liu, D., He, X., and Wang, C. (2021). A visual perception method for autonomous navigation at sea based on ros. In *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, 149–153. doi:10.1109/ICCASIT53235.2021.9633563.

Valencia, R., Andrade-Cetto, J., and Porta, J.M. (2011a). Path planning in belief space with pose slam. 78–83. doi:10.1109/ICRA.2011.5979742.

Valencia, R., Andrade-Cetto, J., and Porta, J.M. (2011b). Path planning in belief space with pose slam. *Proceedings - IEEE International Conference on Robotics and Automation*, 78–83. doi:10.1109/ICRA.2011.5979742.

Villa, J., Vallicrosa, G., Aaltonen, J., Ridao, P., and Koskinen, K.T. (2020). Model-based guidance, navigation and control architecture for an autonomous underwater vehicle. In *Global Oceans 2020: Singapore – U.S. Gulf Coast*, 1–6. doi:10.1109/IEEECONF38699.2020.9389247.

Xu, Y., Zhang, T., and Bao, Y. (2021). *Analysis and Mitigation of Function Interaction Risks in Robot Apps*, 1–16. Association for Computing Machinery, New York, NY, USA. URL `https://doi.org/10.1145/3471621.3471854`.

Xuexi, Z., Guokun, L., Genping, F., Dongliang, X., and Shiliu, L. (2019). Slam algorithm analysis of mobile robot based on lidar.

Yang, L. and Chi, H. (2021). Slam self - cruise vehicle based on ros platform. In *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, 6–11. doi:10.1109/ICCECE51280.2021.9342204.