SEVA3D: Autonomous Vehicles Parking Simulator in a three-dimensional environment

MILTON ROBERTO HEINEN[1]
FERNANDO SANTOS OSÓRIO[2]
FARLEI JOSÉ HEINEN[2]
CHRISTIAN KELBER[3]

UNISINOS - Universidade do Vale do Rio dos Sinos
PIPCA - Computao Aplicada
CEP 93022-000 So Leopoldo (RS)
[1]mheinen@turing.unisinos.br
[2](fosorio,farlei)@unisinos.br
[3]kelber@eletrica.unisinos.br

Abstract. This paper describes a simulation system proposed in order to study and to implement intelligent autonomous vehicle control. The developed system can automatically drive a vehicle, implementing a robust control system capable of simulating in a realistic way autonomous parking in a parallel parking space. The system controls the vehicles based on the reading of sonar sensors and automatically generates acceleration and steering commands, parking it in a parallel parking space. The controller was implemented using a rule-based finite state automaton, and the results obtained in our simulations demonstrated that the proposed controller is perfectly able to correctly park the vehicle in different situations.

Keywords: Mobile robots, autonomous robots, intelligent sensorial-motor control, autonomous drive and park.

(Received August 7, 2006 / Accepted October 27, 2006)

1 Introduction

The autonomous vehicles and robots (Autonomous Mobile Robots - AMR) have attracted the attention of a great number of researchers, mainly due to the great challenge that this new domain of research offers: to endow these systems with an intelligent reasoning capability, exploring their abilities to interact with the environment where they are inserted [2]. AMRs should perceive the environment through their sensors (e.g. infrared, sonar, lasers, video cameras), and from them they can be able to plan and execute their actions [3].

Nowadays, mobile robots are used in different areas, as for example, to disarm bombs, to explore hostile environments like volcanos and other planets, and to transport materials, working as (semi) autonomous industrial robotic vehicles. Some well known examples of successful AMRs are: the ALVINN and the newer NavLab systems developed at the CMU's NavLab [17, 1] that guided an autonomous vehicle across the American highways; the NASA's rover robots sent to explore Mars [18]; the robot Dante that is able to explore the interior of volcanos [13]; the different vehicles developed to participate in the Grand DARPA Challenge of Autonomous Ground Vehicles; and the control system of an electric vehicle developed at INRIA in France [16, 12]. All those systems possess the capacity to receive sensor inputs giving to them information about the external environment, and then they generate action commands in a semi or completely autonomous way. They are able to move across the environment in a safe way, or in other words, not colliding against obstacles or risking their integrity or the integrity of the different elements present in the environment.

Starting from the studies and research work developed by the Artificial Intelligence Group (GIA)[1]) and

[1]GIA-PIPCA – http://www.inf.unisinos.br/gia-pipca.html

by Autonomous Vehicles Group (GPVA)[2]) at our institution, Unisinos, related to the development of applications in the area of mobile autonomous robots, the bases of this work were created. We stand out particularly the initial development of the SEVA2D system (Autonomous Vehicles Parking Simulator) [15] that accomplished a simple simulation of the task of parking a non-holonomic mobile robot (car like) in a parallel parking space. This simulator uses a bidimensional environment (2D) in which a simple virtual vehicle with six infrared sensors is controlled through a finite state automaton (SEVA2D-A) or through an artificial neural network (SEVA2D-N) called *Jordan Cascade Correlation Net* (J-CC) [15].

One of the main limitations of the initial version of SEVA is due to the fact it was adopted a simple bidimensional simulated environment, in which the objects are flat. The simulated model of sensors was very simple since the measured distance from objects can't consider their height or tridimensional shape. In a three dimensional simulated environment (3D), as in the real world, tasks such as to localize the sidewalk's curb are more difficult to be executed. In this specific case, curbs usually possess a very small height, measuring 15cm on average. Other important limitation of the previous 2D model was the absence of noise in data read from the simulated infrared sensors. The sensor simulation was very simple, measuring precisely the distance from sensors to well defined obstacles. This model proved to be too much simplified related to the reality.

Due to these limitations, it was proposed the development of a new system, the SEVA3D [9, 8] (Autonomous Vehicles Parking Simulator in a three-dimensiona environment)[3]. The main goal was to develop a new system capable of accomplishing virtual and realistic 3D simulations of vehicles. The same task of controlling a vehicle in order to autonomously park in a parallel parking space should be accomplished by this new system, but instead of using a simple 2D model, it was adopted a more realistic three-dimensional environment model with a 3D sonar sensor model, which includes noise as in real sonar data. Therefore SEVA3D is much closer to the reality, and the simulation results will be easily transposed to our real working system: an automated Mini-Baja Buggy developed by the GPVA Group at Unisinos[10].

The SEVA3D system uses the SimRob3D simulation library and components [7]. The control system implemented in SEVA3D accomplishes the parking of vehicles, controlling them in an autonomous way using a finite state automaton (FSA). The sonar sensors were installed in strategic positions and orientations around the vehicle, making possible to control the vehicle in the parking task. This paper aims to describe the modeling, development and implementation of SEVA3D. In the following sections it will be described: the adopted simulation model, the vehicle controller based on a finite state automaton, the accomplished experiments and obtained results. We finish presenting some possible improvements and future work.

[2]GPVA – http://www.eletrica.unisinos.br/˜autonom/

[3]http://inf.unisinos.br/˜osorio/seva3d/

2 Related Works and Main Concepts

Studies relative to assistive systems and (semi) autonomous parking of vehicles have been done by some research groups and companies (e.g. BMW, Mercedes-Benz, Toyota, GMC) in the US, Japan, UK, Italy, Germany and France. Among them, one of the first outstanding studies was accomplished by INRIA researchers [16, 12, 5], which implemented a control system used to park an adapted Ligier electric vehicle in an autonomous way. This vehicle was equipped with 14 sonar sensors and it used a manually customized set of rules to accomplish the task. The Figure 1 shows the INRIA adopted vehicle and the parking model.

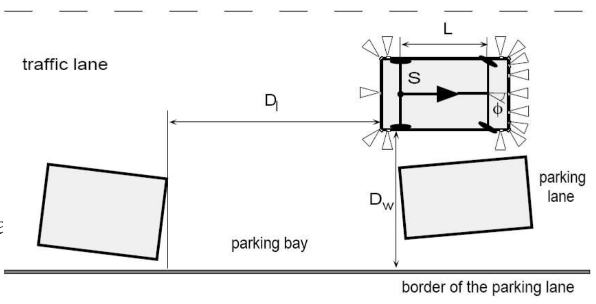


Figure 1: INRIA's vehicle parking model [12]

The above mentioned system adopted the Ackerman kinematics model [3] to describe the vehicle control path. The parking task was divided in three stages: to search for a parking space location; to adjust the vehicle position to allow a correct parking; to control the parking manoeuvre execution. While the vehicle moves along the road searching for a vacant place, an environment map is elaborated considering the sensorial input data. When a parallel parking space with enough size is located, the system estimates the beginning manoeuvre position and moves the vehicle up to this position. Once the vehicle is correctly positioned, the parking manoeuvre starts. This task is accomplished through the use of sinus based functions, as described in [14]), defining a soft path, as showed in Figure 2.

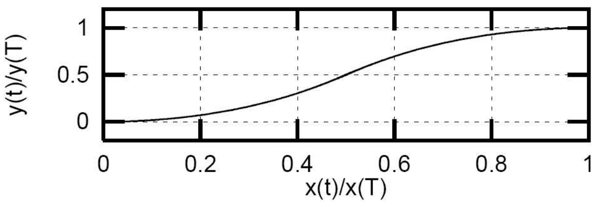


Figure 2: Parking path curve [12]

During all parking manoeuvre execution, the vehicle is controlled in an autonomous way using an iterative algorithm, and for each iteration, the speed and the steering wheel angle are adjusted using the following equations:

$$\phi(t) = \phi_{max}\, k_\phi\, A(t), \quad 0 \le t \le T \tag{1}$$

$$v(t) = v_{max}\, k_v\, B(t), \quad 0 \le t \le T \tag{2}$$

where $\phi(t)$ is the angle of the steering wheel at the instant t, $v(t)$ is the speed of the vehicle at the instant t, T is the maximum duration of the parking manoeuvre, ϕ_{max} is the maximum steering angle, and v_{max} is the maximum speed during the manoeuvre. The $k_\phi = \pm 1$ indicates if the parking space is on the left (-1) or on the right $(+1)$ side, and $k_v = \pm 1$ indicates the direction of the movement, assuming the value $+1$ to move forward and -1 to move backward. The values of $A(t)$ and $B(t)$ are estimated through the following equations:

$$A(t) = \begin{cases} 1, & 0 \le t < t', \\ \cos \frac{\pi(t-t')}{T^*}, & t' \le t \le T - t', \\ -1, & T - t' < t \le T, \end{cases} \tag{3}$$

$$B(t) = 0.5(1 - \cos \tfrac{4\pi t}{T}), \quad 0 \le t \le T, \tag{4}$$

$$t' = \tfrac{T - T^*}{2}, \quad T^* < T, \tag{5}$$

where T (duration of the manoeuvre) and T^* (duration of the curved part of the manoeuvre) are estimated based on the width (Di) and depth (Dw) of the parallel parking space.

In order to become possible to measure in advance the parking space depth, it was necessary to install a barrier of moderate height close to the curb. The introduction of this artificial barrier turns possible to determine more precisely the parking space depth using the available sensors [16]. One of the advantages of this approach was the choice of sinus based functions to control the vehicle, producing softer movements, instead of using a finite state automaton based on discrete states, which can cause abrupt state transitions and action changes. The drawbacks of this approach are: (i) the requirement of curb barrier installation, which restricts the practical application of this method on conventional streets; (ii) the great number of sonar sensors that must be installed around the vehicle (fourteen sensors); (iii) the limited application of this algorithm that works specifically in parallel parking tasks and needs to be manually coded. We aim to propose a new system that uses a small number of sensors, should be able to be used in conventional parking spaces, and also must allow a simple adaptation to different parking situations. The SEVA3D is the system we developed and we are improving in order to achieve these goals.

3 SEVA3D Simulator

The SEVA3D Simulator possesses several improvements related to the original SEVA (2D), among which it can be emphasized the following main features:

- Uses a 3D environment and 3D simulated sensors obtaining more realistic simulations;
- Implements sonar sensors similar to real sensors, including the presence of noise in the model;
- Accomplishes the autonomous vehicle control, parking in a parallel parking space independently of the presence or absence of other previously parked cars;
- Implements a robust control system accepting different starting positions, with a distance between the car and the curb ranging from 2 to 4 meters;
- Accomplishes automatic adjustments of the vehicle position, if the vehicle is too close (or too far) from the parked cars, doing this before beginning the automatic parking manoeuvre (considers too close when the side by side distance is smaller than 30cm);
- Allows the visualization of the parking manoeuvre from virtually anywhere in the 3D virtual environment;
- Accomplishes the reversal parking manoeuvre, exiting the vehicle from the parking space in an automatic way.

The SEVA3D was able to correctly control the vehicles in parking manoeuvres, as presented in experimental results section. The Figure 3 shows a diagram of the SEVA3D modules. The main components of the SEVA3D simulator are:

Perception module: measures the distance from obstacles based on a sonar sensor simulation (see Section 3.2);

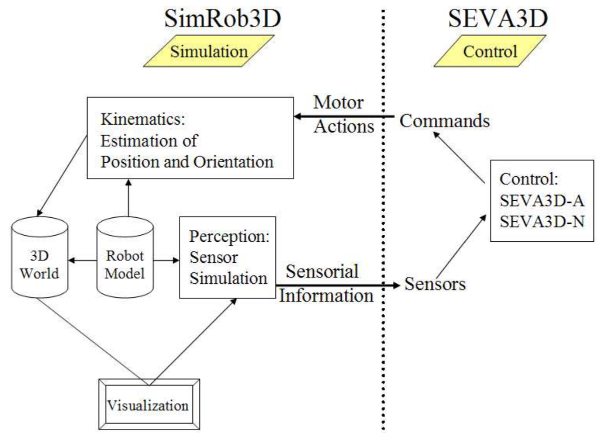


Figure 3: SEVA3D modules

Action module: sends action commands to vehicle actuators, defines the direction (forward, backward), the speed (gas pedal control) and the car orientation (steering wheel rotation) (see Section 3.4);
Kinematics module: uses the Ackerman model to estimate the vehicle trajectory (simulation), considering vehicle direction, speed and steering wheel angle defined by the Vehicle Action Module (see Section 3.3);
FSA control module: receives sensor data from the perception module and sends actions to the action module, using a Finite State Automaton to control the parking manoeuvre (see Section 3.5) and pull-out manoeuvre (see Section 3.6).

3.1 SimRob3D Simulation Tool

The implementation of SEVA3D uses the SimRob3D simulation tools [7], previously developed by our group. The SimRob3D tools main characteristics are: to provide 3D environment visualization tools used in simulations, and to provide customizable mobile robots simulation tools. The 3D environment objects can be modeled using different commercial or free 3D modeling software, and allow to detail the different elements present in the environment (objects - vehicles, streets, buildings; lights and textures), resulting in a high level of realism in the simulations. The SimRob3D tools also include different sensorial (infrared and sonar sensors) and kinematics models (Ackerman and Differential models), which can be used to customize the simulated robots.

SimRob3D tools provide only a set of methods used to read data from sensors and to send commands to actuators, updating the robot position in the simulated environment. SimRob3D needs an external robot control module, since their main goal is only to provide facilities to model the robots (defining sensors and actuators) and to provide a simulated interface between the robots and the environment. The external controller is a separated module from SimRob3D, in our case provided by SEVA3D. The interface between these two softwares is accomplished by DLL (dynamic linked library) calls. So, the SimRob3D API (Application Programming Interface) supplies an interface that separates world interaction from the robot control mechanisms. SEVA3D uses SimRob3D facilities to define the environment, vehicle, sensors and actuators, implementing its own vehicle control modules (FSA control module). An interesting characteristic of this coupled system is that the controller does not possesses direct access to any information from the environment, the only available information is the one provided by the vehicle sensors through DLL calls.

3.2 Sensors Model

Sonars are an interesting type of distance sensor used in robotic applications, because they can estimate with a reasonable precision the distance from objects positioned near to them. Sonars can be used to perceive the environment objects and obstacles, and they offer a good precision/price rate related to other distance measuring methods.

The simulated sonar sensors [3] allow to estimate the distance between the vehicle and the obstacles present in the environment: other cars and the edge of the sidewalks. The five sensors used were distributed in strategic positions of the vehicle, as showed in Figure 4.

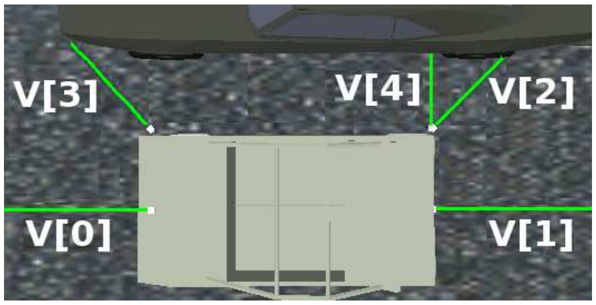


Figure 4: Distribution of the sonar sensors around the vehicle

In our experiments sensors were put only in one side of the vehicle, because all experiments were developed in order to park the vehicle in parallel parking spaces located on the right side of the car, which is a typical situation in two-way streets. Sensors **V[2]** and **V[3]** were positioned with a certain inclination related to the ground, so it was possible to detect the curb. The SEVA3D allows to configure other distributions of the sensors, but in this paper the discussion is focused only to this kind of vehicle configuration.

The SimRob3D sonars are simulated through the definition of a conical section of the virtual space, where the objects which remain inside this volume can be detected. The intersection between objects and the sonar cone volume (perceptual space) is detected using a stochastic approach. Several object detection lines (rays) are generated from the position of the sensor directed according to the sonar spatial orientation, remaining inside the sonar cone volume. A RayCast[4] [4] technique was used to generate rays, that are randomly distributed in the sonar cone volume. If any of them collide (intersect) with some object polygon, the distance from the sensor until the collision point is informed.

Due to the stochastic nature of the measured distances, a temporal window method [6] was adopted to represent the data obtained from sensors. The defined temporal window has a length of 10 samples. This technique is also commonly used to deal with real sonar sensors data.

Besides the sonar sensors, sometimes it was also necessary to use an odometer. The odometer was used only to verify if the parking space is enough to allow the correct parking of the vehicle when there are no other parked cars available to be used as reference points. This situation occurs only when there are large empty spaces.

3.3 Kinematics Model

The movement of the vehicle respects the Ackerman kinematics model [3], which was also the model adopted in the precursor work developed at INRIA [5]. In this model a simulated vehicle is represented by a rectangular volume supported by four wheels divided in two axes, where the back wheels are attached to a fixed axis and the front wheels can be turned, controlling them through the steering wheel. The Figure 5 shows the elements of the kinematics model.

The vehicle's location (position and orientation) relative to some reference coordinate system is denoted as $q = (x, y, \theta)^T$ where $x = x(t)$ and $y = y(t)$ are the coordinates of the rear axle midpoint, $\theta = \theta(t)$ is the orientation of the vehicle, and t is time. The motion of the vehicle is described by the following equations [12]:

$$\begin{cases} \dot{x} = v\,\cos\phi\,\cos\theta, \\ \dot{y} = v\,\cos\phi\,\sin\theta, \\ \dot{\theta} = \frac{v}{L}\,\sin\phi, \end{cases} \tag{6}$$

where $\phi = \phi(t)$ is the steering angle, $v = v(t)$ is the locomotion velocity of the midpoint of the front wheel

[4]RayCast is a computer graphic technique that simulates the physical effects associated with the propagation of light rays [4]

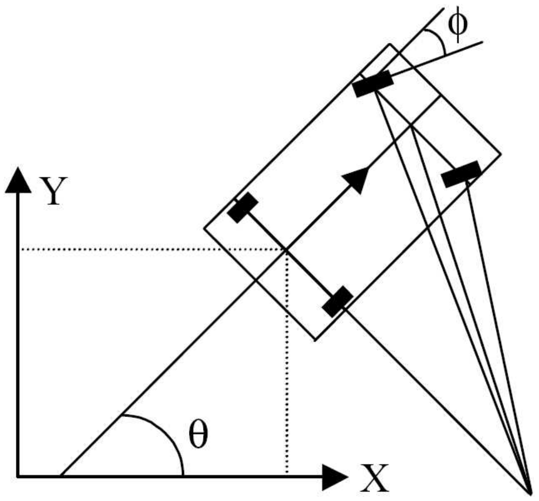


Figure 5: Kinematics model

axle, and L is the wheel base. The steering angle and locomotion velocity are two control commands (ϕ, v). Eqs. 6 correspond to a system with non-holonomic constraints because they involve the derivatives of the coordinates of the vehicle and are non-integrable [11].

3.4 Vehicle Control

The vehicle is controlled in SEVA3D by simulated actuators. These actuators control the vehicle speed (accelerate/break) and steering wheel rotation. So, the vehicle displacement is obtained in the simulator sending commands to actuators which control the speed (v) and rotation of the steering wheel (ϕ).

Differently of the model adopted in SEVA2D, whose parameters are not directly related to real world measures (distance was measured in pixels), the parameter values adopted in SEVA3D are quite realistic. As for example, during the parking manoeuvre in SEVA3D the speed can be set with values ranging from 0 (stopped) to 80 (very fast). When the vehicle needs to move backward the speed is set to a negative value.

To change the vehicle orientation, the steering wheel angle (ϕ) can be set with values ranging from 0 and 35, that corresponds directly to the rotation angle to the left of the steering wheel. When the vehicle needs to turn to the right the angle ϕ is set to a negative value.

3.5 FSA Control - Parking

In SEVA3D, the vehicle control in an autonomous parking task is accomplished by a Finite State Automaton (FSA). The Figure 6 shows the diagram of the finite state machine used to control the vehicle. The following states were defined:

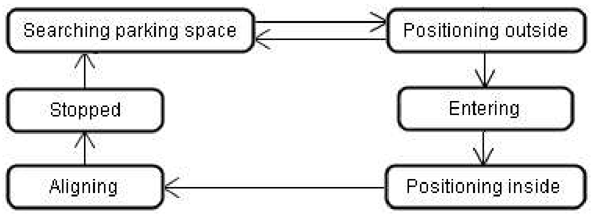


Figure 6: Automaton states

Stopped: automaton initial and final state;
Searching for parking space: first state of the parking manoeuvre, when the vehicle moves in a straight forward direction, searching for a free parking space. If a possible parking space is found the FSA state changes to *Positioning outside*;
Positioning outside: the vehicle moves forward in order to reach a correct position to start entering in the parking space, usually aligning side by side with the next parked car (the odometer can be used in the absence of parked cars). This state is also used to verify if the parking space is adequate. If the space is too small, the state returns to *Searching for parking space*. If the space is large enough, the state changes to *Entering*;
Entering: the car starts to move backward and the steering wheel is turned to the right, so in this way the vehicle starts to enter in the parking space. The ϕ and v values are set to empirically predefined values. When the sensor **V[2]** (Figure 4) detects the sidewalk curb the state changes to *Positioning inside*;
Positioning inside: in this state, the vehicle continues to move backward, but the steering wheel is turned to the left. When the sensor **V[3]** detects the sidewalk curb or the sensor **V[1]** detects a close obstacle (distance under 30cm) the state changes to *Aligning*;
Aligning: in this state, the vehicle is moved in order to reach an adequate distance from the cars parked ahead and behind it. After the alignment is done, the state changes to *Stopped* and the manoeuvre is terminated with success.

3.6 FSA control - pull out

The Figure 7 shows the diagram of the finite state machine used to pull out the vehicle of the parking space. The following states were defined:

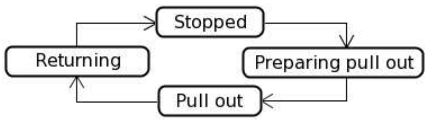


Figure 7: FSA control - pull out

Preparing pull out: the car starts to move in a straight backward direction, until the **V[1]** detects a near obstacle (another parked car) or until the frontal space is sufficient to going out;
Pull out: the vehicle moves in a forward direction executing a "s" shaped trajectory, until the sensors indicate the car is outside of the parking space;
Returning: the car is aligned in a parallel way to the lane, the state changes to *Stopped* and the maneuver is terminated.

4 Implementation

The SimRob3D simulation tools [7] were used to create the virtual environment and to interface the vehicle devices with the SEVA3D autonomous controller implementation. The virtual environment was modeled, creating 3D models of the road and parked cars, and also the model of our autonomous vehicle. The Figure 8 shows an image of the virtual environment modeled.

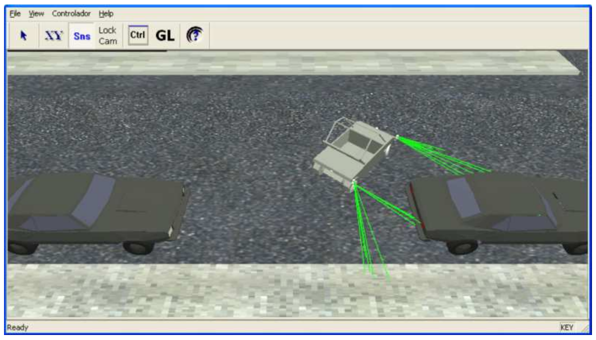
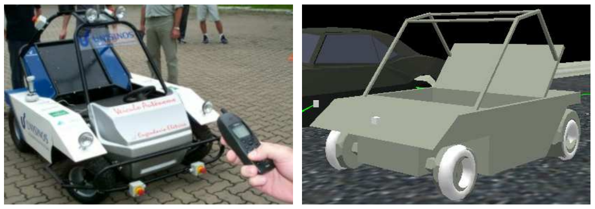


Figure 8: Virtual environment visualization

The model of the vehicle used to accomplish the parking task is a reproduction of a real Mini-Baja Buggy available in our research laboratory. This vehicle was developed by the GPVA Research Group at our institution, Unisinos. The real vehicle was automated and now it can be controlled from remote devices, like cell phones (see available videos in the GPVA web site). The Figure 9 shows the actual vehicle used as model of the virtual autonomous vehicle. At the present time we are working on the real vehicle instrumentation, adding sonar sensors, and a real world test is planned soon using the SEVA3D controller to control the real vehicle.

The SEVA3D simulator implements a discrete integration of the model described in the Eqs. 6. The FSA implementation, used to control the simulated vehicle movement, was developed in "C" language. In order to follow the progress of the parking manoeuvre, the user can visualize the simulated environment using a virtual

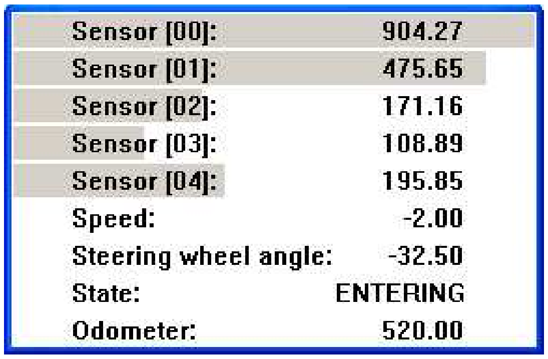


(a) Real vehicle (b) Simulated vehicle

Figure 9: Automated Mini-Maja vehicle

camera. Besides that a status window exhibits information about the simulation, including: the FSA current state, the sensors data, the vehicle speed and steering wheel angle, the absolute vehicle position and orientation, and the odometer value. The Figure 10 shows the window containing all these information.

The SEVA3D implementation was validated through several preliminary tests. It was also verified that the simulated model behavior was quite similar to the reality. In the next section are presented the main experimental results obtained with this system.



Sensor [00]:	904.27
Sensor [01]:	475.65
Sensor [02]:	171.16
Sensor [03]:	108.89
Sensor [04]:	195.85
Speed:	-2.00
Steering wheel angle:	-32.50
State:	ENTERING
Odometer:	520.00

Figure 10: Status information window

5 Experimental Results

In order to validate the SEVA3D autonomous parking control system, several experiments were planned and realized. To verify the system robustness, different simulations were accomplished with success, changing the configuration of the following items in each simulation:

- The initial position of the autonomous vehicle in the street. Different initial positions were tested varying from few centimeters to more than 3 meters distant from the parked cars;
- The parking manoeuvre was tested in the following situations: between two cars, in the presence of just one parked car and without any other previously parked car in the street;
- The parking manoeuvre was also tested close to corners and garage entrances, where the curb reference point doesn't exist in front of (or behind) the parking space.

In another series of experiments we evaluated the final vehicle position as evidence of a successful parking manoeuvre execution. Due to the stochastic nature of the sensorial data, 10 experiments were achieved using different random seeds in each simulation. When the simulations were finished the average was calculated and the standard deviation of the vehicle distance from the curb. In all these experiments, SEVA3D was capable to correctly park the vehicle, with an average distance from the curb of 26.16cm and a standard deviation of 5.92cm. This demonstrates that the implemented system is safe and robust to control vehicles in parallel parking tasks execution. The Figure 11 shows an example of parking manouevre[5].

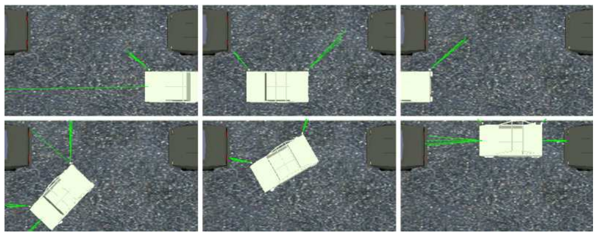


Figure 11: Parking maneuver

6 Conclusions and Perspectives

This work main goal was to develop a simulator for autonomous control of vehicles in parallel parking tasks. The proposed system should be able to create a realistic model of the real world application, so it was implemented a simulation tool situated in a 3D environment, the SEVA3D. This system includes a 3D model of the vehicles and obstacles, and it also includes a 3D model of sonar sensors. The experimental results, accomplished with SEVA3D, demonstrated that the control system possesses the capacity to correctly control the vehicle. The main objective of the control system was achieved with success: to park vehicles in an autonomous way, not colliding against obstacles present in the environment. In order to validate the SEVA3D vehicle control module, several experiments were accomplished with visual and numerical evaluations. This

[5]Some videos demonstrating the SEVA3D parking manoeuvre are available in http://www.inf.unisinos.br/~osorio/seva3d/

experiments allowed to verify that the vehicle was correctly controlled in different situations, demonstrating that the proposed method is stable, safe and robust.

Although the experimental results were very good, we are still planning to improve SEVA3D/SimRob3D simulation model. The implementation of a new version of SEVA3D is being considered in order to include a more realistic physical simulation tool. We are considering to add rigid body dynamics simulation extensions, allowing to simulate force, torque, friction, gravity, terrain slopes, etc. In the near future we plan to implement SEVA3D in a real vehicle, an automated mini-Baja Buggy available in our research laboratory. The SEVA3D will be adapted to use the new hardware (real sonar sensors) and the system will be evaluated in real world conditions.

References

[1] Batavia, P., Pomerleau, D., and Thorpe, C. Applying advanced learning algorithms to alvinn. Technical Report CMU-RI-TR-96-31, Carnegie Mellon University (CMU), Pittsburgh, PA, 1996.

[2] Bekey, G. A. *Autonomous Robots: From Biological Inspiration to Implementation and Control*. The MIT Press, Cambridge, MA, USA, 2005.

[3] Dudek, G. and Jenkin, M. *Computational Principles of Mobile Robotics*. Cambridge University Press, Cambridge, UK, 2000.

[4] Foley, J. D. *Introduction to Computer Graphics*. Addison-Wesley, xxviii, 1994.

[5] Garnier, P., Fraichard, T., Laugier, C., Paromtchik, I., and Scheuer, A. Motion autonomy through sensor-guided manoeuvres. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyongju, Korea, Sept. 1999, IEEE Press.

[6] Haykin, S. *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, Upper Saddle River, NJ, USA, 2 edition, 1999.

[7] Heinen, F. J. and Osório, F. S. HyCAR - a robust hybrid control architecture for autonomous robots. In *Proceedings of Hybrid Intelligent Systems (HIS)*, volume 87, pages 830–840, Santiago, Chile, 2002. Soft Computing Systems, IOS Press.

[8] Heinen, M. R., Osório, F. S., Heinen, F. J., and Kelber, C. Autonomous vehicle parking and pull out using artificial neural networks. In *Proceedings of the I Workshop on Computational Intelligence (WCI)*, Ribeirão Preto, SP, Brazil, Oct. 2006. International Joint Conference IBERAMIA/SBIA/SBRN 2006, IEEE Press.

[9] Heinen, M. R., Osório, F. S., Heinen, F. J., and Kelber, C. SEVA3D: Using artificial neural networks to autonomous vehicle parking control. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, Vancouver, Canada, July 2006. IEEE World Congress on Computational Intelligence (WCCI), IEEE Press.

[10] Kelber, C., Jung, C. R., Osório, F. S., and Heinen, F. J. Electrical drives in intelligent vehicles: Basis for active driver assistance systems. In *Proc. of IEEE International Symposium on Industrial Electronics (ISIE)*, volume 4, pages 1623–1628, Dubrovnik, Croatia, 2005. IEEE Press.

[11] Latombe, J. C. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, USA, 1991.

[12] Laugier, C., Fraichard, T., Paromtchik, I. E., and Garnier, P. Sensor based control architecture for a car-like vehicle. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 165–185, Victoria, Canada, Oct. 1998.

[13] Lemonick, M. Dante tours the inferno. *Time Magazine - Time Domestic/Science*, 144(7), 1994.

[14] Murray, R. and Sastry, S. Steering nonholonomic systems using sinusoids. In *Proc. of 29th IEEE Int. Conf. on Decision and Control (CDC)*, pages 2097–2101, New York, USA, Dec. 1990.

[15] Osório, F. S., Heinen, F. J., and Fortes, L. Autonomous vehicle parking using finite state automata learned by J-CC artificial neural nets. In *Proceedings of the VI Brazilian Symposium on Neural Networks (SBRN)*, volume 1, Porto de Galinhas, PE, Brazil, 2002. IEEE Press.

[16] Paromtchik, I. E. and Laugier, C. Autonomous parallel parking of a nonholonomic vehicle. In *Proc. of IEEE Int. Symposium on Intelligent Vehicles (IV)*, pages 13–18, Tokyo, Japan, Sept. 1996.

[17] Pomerleau, D. A. Neural network based autonomous navigation. *Vision and Navigation - The CMU Navlab*, 1990.

[18] Stone, H. W. Mars pathfinder microrover - a small, low-cost, low-power spacecraft. In *Proceedings of AIAA Forum on Advanced Developments in Space Robotics*, Madison, WI, Aug. 1996. American Institute of Aeronautics and Astronautics (AIAA).