estimation from other sensors allows the required robustness to GPS signal losses [8].

The process of Simultaneous Localization Mapping and Moving Objects Tracking is called SLAMMOT.

A. SLAMMOT

As the main sensor to detect obstacles we use a laser scanner. We suppose that the surrounding of the vehicle is locally planar (strong assumption). The laser scanner provides angle and distance measures to obstacles in a radius of 40 $[m]$ with a frequency of 20 $[Hz]$. Its measures provide information about the occupied space and the free space.

A laser scan is a set of l measured points x_i. For each new scan each measure is associated to a weight w_i reflecting the a priori probability that the point x_i is measuring a static object. Using this weights factors, the set of measures is matched with the current map. Based on the coherence between free, occupied and unobserved space, measures of moving objects are detected [9]. Since moving objects measures are separated from static objects measures, the detection of moving objects (grouping of measures) and they tracking becomes a simpler task. The different objects are classified based on their geometric properties and their motion (small round elements are pedestrians, large rectangles are cars, etc...).

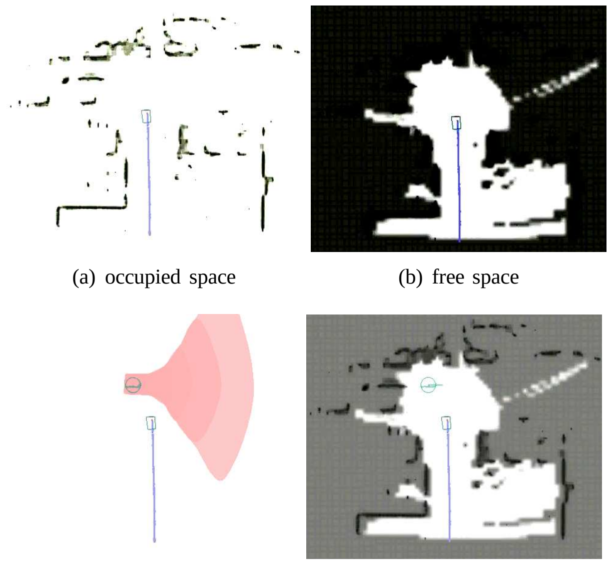
In order to keep a lightweight representation that allows fast scan processing and a correct management of the uncertainty, the world model is decomposed in three interrelated elements: a grid of gaussians of the occupied space [10], a first order interpolated grid of the free space, and a set of tracked moving objects (see figure 1). The tracked objects, the current maps and the uncertainty in the displacement between two laser scans are used to estimate the initial weights w_i.

For more details on this algorithm, the reader can consult [5].

Due to the ever changing nature of the urban environment, and since the vehicle is not expected to pass through a streets immediately after leaving it, the perception module does not need to build and store a detailed map of the city. It only keeps a short term memory of the surroundings as required for the trajectory planning task.

B. Ground traversability

The laser scanner employed is a 2D sensor, installed horizontally. In such setup, the sensor does not provide information about the ground (or low obstacles in general). Some other works use 3D measures from 3D scanners or by using a myriad of parallel 2D scanners. 3D laser scanners are too slow for measurements in motion, 3D ladars [11], [12] are still a technology in development and using multiples 2D sensors in parallel was considered costly for our application. Even with a detailed 3D map of the area to traverse, colour information is required to detect grass or ground landmarks indicating undesirable or prohibited areas (laser scanner intensity measures can be used for this task).



(a) occupied space (b) free space

(c) detected moving object and conservative prediction (d) SLAMMOT output

Figure 1. Different components of our SLAMMOT algorithm internal representation and final output

In order to determine the ground traversability we complement the 2D information of the laser with a wide angle colour video camera. To keep the solution as generic as possible we use a machine learning approach to process the image frames. Each pixel of the image will be classified as traversable or not. The traversability is defined on the basis of training examples fed previously manually.

The input image is trivially split into a set of small rectangular regions, that are processed individually. Each small region is transformed into a features vector (with 40 elements, as defined in [13]) describing the region texture. In order to lower the on-line classification computation cost, we use an “automatic relevance determination” (ARD) kernel with an “Informative Vector Machine” (IVM) [14] to select the most relevant features given the training set. Then we train a “Relevance Vector Machine” (RVM) [15] with the trimmed features vectors. In our tests sets, the RVM provides similar classification rates (superior to 95%) than the IVM and the classic SVM [16] while using less than half of the relevance/support vectors; the on-line classification speed is more than two times faster than SVM (at the cost of a slower learning rate). Figure 2 illustrates our typical training examples and the output of the classifier on new images.

Once each region is classified, the resulting binary image can be projected to the ground plane using the camera calibration (internal and external parameters). This binary measure should be used to fed an occupancy grid estimating the probability of a region to be or not traversable (since we expect 5% of misclassifications).

Due to computation limitations in our current implementation the largest and lowest region of the binary image is selected as the ground region, the enclosing polygon is

(a) One of the training examples

(b) Classification results examples

Figure 2. A simple machine learning approach is used for road detection

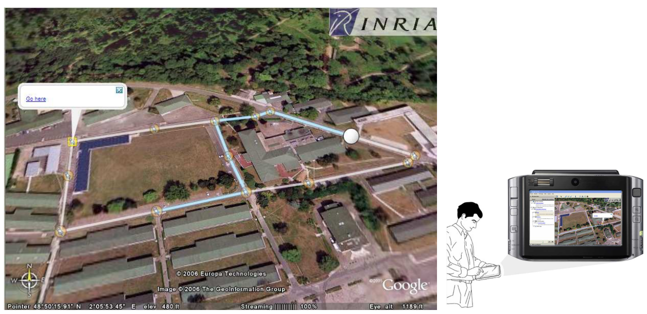


Figure 3. Left: route planner interface, indicating current vehicle position, planned route, and allowing to change the destination. Right: illustrating an user doing a request for the vehicle

projected to the ground and used as additional static obstacles in the world model. With each new image, the polygon delimiting the road is updated. Since the vehicle is large compared to the small errors in the polygon, this gross approximation seems good enough when the road surface has a simply connected geometry (strong assumption). Using a simply connected polygon instead of the dense grid map imply that we trade off computation cost to managing situations such as holes in the road.

IV. Route planning

Since the road network of our campus is small enough and its traffic is negligible a simple Dijkstra is good enough for our application. For route planning on large networks without considering traffic predictions methods such as the one available on commercial navigation system should be used. Route planning on city wide networks considering traffic predictions (in a scenario where every agent is informed and rational) seems to be a variant yet to be explored.

The route planner runs on a separate server with an end used interface rendered through Google Earth (see figure 3). The route server is periodically communicating each few seconds with the vehicle's software to update the route and retrieve the current location of the vehicle.

V. Trajectory planning

When evolving in an populated (pedestrians and cars), uncertain (noise in the observations) and incomplete (partially observed) world only limited guarantees can be given on the harmless motion of the robot. In [6] we show that it is possible to guarantee that the robot will not harm by action if at anytime it provides a trajectory able to stop without colliding (or entering into a non traversable area). In case of collision, the vehicle is guaranteed to be with null velocity with respect to the ground. It can also be shown that if every mobile respects this criterion, then no accidents would arise.

Providing guarantees of not harming by inaction (while the vehicle is stopped) is still an open problem in the general case.

In order to respect the motion constraints of the vehicle, we use a model of the vehicle capabilities and formulate the planning problem as a search problem in the commands space. When the sequence of commands p is executed at state $x(t_1)$ the vehicle will follow the trajectory $\pi(x(t_1), p)$ (given that the vehicle model is correct, and that the controller respect the predicted bound). We search then the plan p that will move the vehicle along the defined route while avoiding reaching any state where $h(x(t_2), t_2) \neq 0$ and where the final state has null speed.

The search for the best trajectory is executed periodically. In this partial motion planning approach [17] at each iteration the planner will use an updated world model and extend the previous plan. To guide the exploration in the sequence of commands space we use an hybrid between greedy search and RRT method [18], mixing directed search with random search.

Notice that the function h is not included in the cost function used to select the best plan p because safety can not be trade-off. Our cost function over p includes how much of the planned route is traversed (effectiveness) and how smoothly this is done (comfort).

With the iterative nature of the planning approach, the constraint of plans with final state with null speed does not imply that the vehicle will actually stop. In pratice this safety constraint will affect the speed of the vehicle, which is adapted to allow collision avoidance in any case considered by the conservative prediction of the perception module. Following the preference of the cost function the vehicle will try to avoid obstacles and advance towards the goal instead of stopping. Stopping is only done if no other option exist, no other plan was found or because it is the optimal solution for the given the cost function.

Figure 6 includes an example of the typical output of the trajectory planner. See section VII for the description.

VI. Control

In [5] we suggested the use of a naive (proportional) non linear controller. We then define a bound on the maximum tracking error (for the planning stage), and put a on-line watchdog over this error. Any overflow on the tracking error generates an emergency stop.

This naive controller provides convergence to zero error for a constant reference state but does not converge to zero when the reference state change in time (as it is usually the

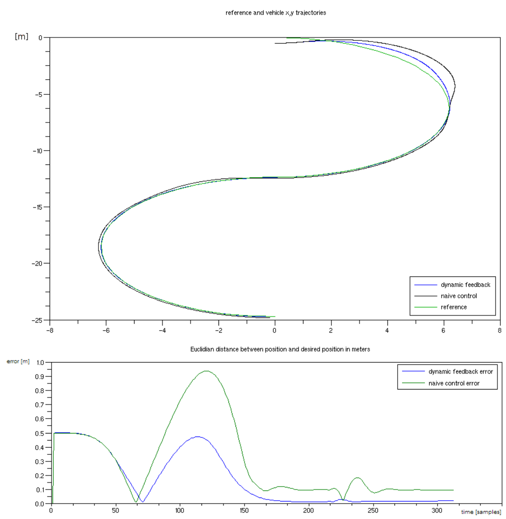


Figure 4. Comparison of the naive and dynamic feedback controllers on an error free situation. The later performs notoriously better

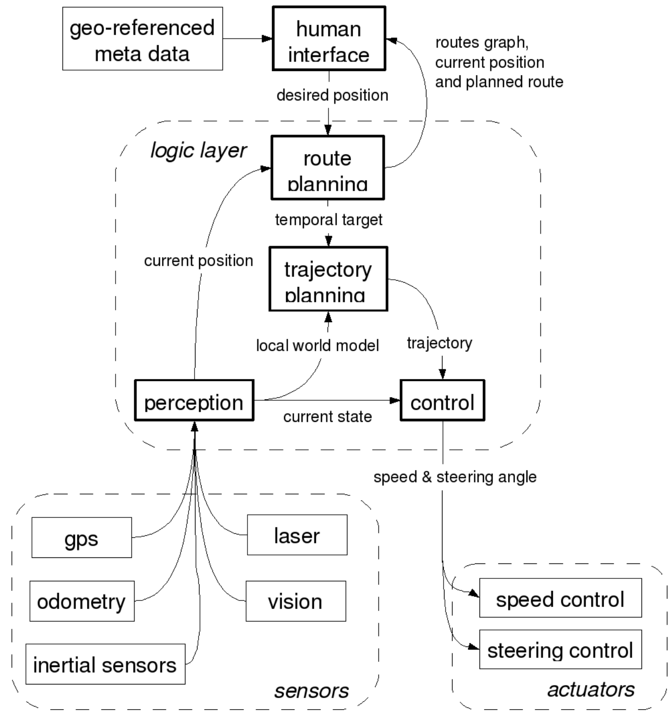


Figure 5. Logic diagram of the system

case). We now use a dynamic feedback controller [19] that provides a theoretical convergence for any feasible trajectory.

In the figure 4 both controllers are compared when following an S shaped trajectory, starting with an initial position error and disregarding noise and model errors. We see that the naive controller does not converge to the constantly changing reference, while the new controller does. The increment on the tracking error (around sample 60) is due to the saturation of the steering angle of the vehicle, which is not modeled by the controllers.

VII. Integration results

The route planning, trajectory planning, perception and control algorithms where integrated into a single system. The logical relation between the different components are presented in the figure 5.

The route planner runs in a separate server communicating to the vehicle through the network. The other modules are executed by a single multi-threaded application written in C++. The laser scanner runs at 15 $[Hz]$ and defines the world model update frequency (a thus the control loop frequency). The image processing runs on a separate low priority thread, grabbing frames as soon as the previous one was processed. With the perception, planning and control running on a dual core 3 $[GHz]$ processor, the 640x480 pixels images processing runs at 2 $[Hz]$. Since image processing is used for ground traversability estimation, and planning is done considering unobserved areas, low frequency on images processing does not imply higher risk. As discussed in section V the vehicle will execute plans where the vehicle will stop before reaching the unobserved areas. In the worst case the scenario, if image processing stall, the vehicle will smoothly stop waiting for the next image to be processed.

Trajectory planning is done using an any time interruptible method with safety guarantees [17]. We fixed the plan update frequency to 2 $[Hz]$, which seems to be a good trade-off between reactivity and planning horizon.

Figure 6 illustrate the internal representation of the vehicle software while running. In this example route planner has sets the target at the top of the image. Gray indicates non observed areas, black detected static obstacles, green circles are the detected moving objects, and the line going out of such circle they estimated direction. The green rectangle indicate the estimated position of the vehicle, the dark line below it its previous positions, the green line on top of it the positions of its planned trajectory. Notice that the plan can pass over moving obstacles since, given the used prediction model, the moving objects are expected (with high certainty) to have leaved that area before reaching it.

Figure 7 presents a sequence of images of a similar situation. At the beginning the vehicle target is set near the photographer position. The vehicle will automatically map the environment, avoid obstacles, adjust the speed to consider the possible apparition of pedestrians on the unobserved area, and finally manage to avoid any collision while reaching the desired position.

VIII. Conclusion and future works

We have described the design and implementation of a driverless vehicle system. The proposed approach provides strong safety guarantees. The vehicle explicitly consider the unknown to adapt the driving speed and trajectory. To our knowledge this is the first driverless vehicle for urban environment that respects the safety constraints presented in [7] and [6] . Most previous systems either fail to explicitly

(a) starts exploring an unknown environment

(b) avoids known obstacles and adapts speed to posible obstacles

(c) stops in front of pedestrian

(d) continue moving

Figure 7. Experiment results. See text section VII

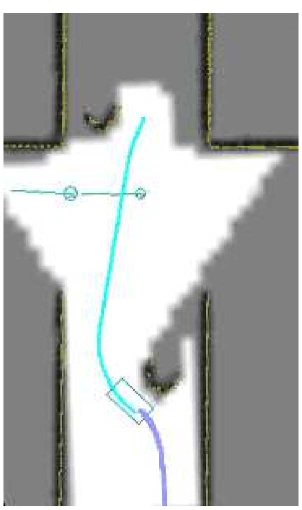


Figure 6. Example of safe planning in a perceived environment

model the moving objects [1], do not guarantee that collision free trajectories will be found in the future [4], or do not use conservative predictions of the possible harm (e.g. enforcing straight lines trajectories to all observed cars).

This system provides the strict minimum for driving capabilities; it can easily integrate more sensors, information communicated from other vehicles or entities, or integrate arbitrary driving rules (as cost functions over trajectories).

Future works will consider creating a vision based dense 3D SLAMMOT perception algorithm, to eliminate the reliance on the laser scanner and enhance the traversability estimation. We will also explore further accelerating the exploration in the space of trajectories by using an approach similar to [20] (piecewise parametrization of the trajectory and optimization of the parameters via gradient descent).

REFERENCES

[1] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont,

L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, “Stanley: The robot that won the darpa grand challenge,” *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, September 2006. [Online]. Available: http://robots.stanford.edu/papers/thrun.stanley05.html

[2] C. Laugier, S. Petti, D. A. Vasquez Govea, M. Yguel, T. Fraichard, and O. Aycard, “Steps towards safe navigation in open and dynamic environments,” in *Autonomous Navigation in Dynamic Environments: Models and Algorithms*, ser. Springer Tracts in Advanced Robotics Series (STAR), C. Laugier and R. Chatila, Eds. Springer, 2006. [Online]. Available: http://emotion.inrialpes.fr/bibemotion/2006/LPVYFA06

[3] R. Benenson, S. Petti, T. Fraichard, and M. Parent, “Toward urban driverless vehicles,” *International Journal of Vehicle Autonomous Systems, Special Issue on Advances in Autonomous Vehicle Technologies for Urban Environment*, vol. 1, no. 6, pp. 4 – 23, 2008. [Online]. Available: http://hal.inria.fr/inria-00115112

[4] S. Kolski, K. Macek, L. Spinello, and R. Siegwart, “Secure autonomous driving in dynamic environments: From object detection to safe driving.” in *Proceedings of the Workshop on Safe Navigation in Open and Dynamic Environments: Applications to Autonomous Vehicles at the IEEE International Conference on Robotics and Systems*, San Diego, USA, 2007. [Online]. Available: http://teamster.usc.edu/ moradi/iros07/iv_iros07.htm

[5] R. Benenson, S. Petti, M. Parent, and T. Fraichard, “Integrating perception and planning for autonomous navigation of urban vehicles,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006. [Online]. Available: http://hal.inria.fr/inria-00086286

[6] R. Benenson, T. Fraichard, and M. Parent, “Achievable safety of driverless vehicles,” 2008, to appear.

[7] T. Fraichard, “A short paper about motion safety,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007. [Online]. Available: http://hal.inria.fr/inria-00134467/

[8] J. Chao, Y. qi Chen, W. Chen, X. Ding, Z. Li, N. Wong, and M. Yu, “An experimental investigation into the performance of gps-based vehicle positioning in very dense urban areas,” *Journal of Geospatial Engineering*, vol. 3, pp. 59–66, 2001. [Online]. Available: http://www.lsgi.polyu.edu.hk/staff/ZL.Li/vol_3_1/06_chao.pdf

[9] D. F. Wolf and G. S. Sukhatme, “Mobile robot simultaneous localization and mapping in dynamic environments,” *Autonomous Robots*, vol. 19, no. 1, pp. 53–65, July 2005. [Online]. Available: http://www-robotics.usc.edu/ gaurav/Papers/publications.php

[10] P. Biber, S. Fleck, and W. Straßer, “A probabilistic framework for robust and accurate matching of point clouds,” in *26th Pattern Recognition Symposium (DAGM 04)*, 2004. [Online]. Available: http://www.gris.uni-tuebingen.de/publics/staff/Peter_Biber.html

[11] “Prevent subproject usercams,” 2005. [Online]. Available: http://tinyurl.com/542c2j

[12] “Swissranger, a miniature 3d time of flight camera,” 2005. [Online]. Available: http://www.swissranger.ch

[13] A. Saxena, S. H. Chung, and A. Y. Ng, “3-d depth reconstruction from a single still image,” *International Journal of Computer Vision (IJCV)*, August 2007. [Online]. Available: http://make3d.stanford.edu/publications.html

[14] N. Lawrence, J. Platt, and M. Jordan, “Extensions of the informative vector machine,” in *Proc. Sheffield Machine Learning Workshop,*, ser. Springer Lecture Notes on Computer Science, 2005.

[15] M. Tipping, “Sparse bayesian learning and the relevance vector machine,” *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001. [Online]. Available: http://www.miketipping.com/index.php?page=rvm

[16] V. N. Vapnik, *The Nature of Statistical Learning Theory*, ser. Computational learning theory. Springer, 2000.

[17] S. Petti, “Safe navigation within dynamic environments: a partial motion planning approach,” Ph.D. dissertation, Ecole des Mines de Paris, 2007.

[18] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006. [Online]. Available: http://planning.cs.uiuc.edu/

[19] E. Yang, D. Gu, T. Mita, , and H. Hu, “Nonlinear tracking control of a car-like mobile robot via dynamic feedback linearization,” in *Proceedings of Control*, University of Bath, September 2004.

[20] A. Kelly, T. Howard, and C. Green, “Terrain aware inversion of predictive models for high performance ugvs,” in *Proceedings of the SPIE Defense and Security Symposium*, April 2007, pp. 6561–6566. [Online]. Available: http://www.ri.cmu.edu/pubs/pub_5740.html