mPlanner: Robot Motion Planning based on Interaction of Planner and Controller[*]

Thomas Wösch
Corporate Technology
Siemens AG
81730 Munich, Germany
thomas.woesch@mchp.siemens.de

Abstract

Service robots serve and assist human beings sharing a common environment. Therefore fast responding and robust planners generating collision-free motions are required to guarantee a safe and convenient man-machine interaction. In most cases the working environment is partially known or perceived by a sensor system. To reduce planning time, this work presents a motion planner interacting with reactive plan execution systems. Dividing the work space into subspaces appropriate motion plans are determined. Based on an environment model collisions are avoided by interacting with an obstacle avoidance system. Tactile sensors are used to detect collisions of the robot arm, e.g. due to an incomplete knowledge of the environment or enforced by a human touching the robot. Experimental results of our 8 degree of freedom manipulator arm mounted on a mobile platform are presented. In these experiments the robot acts as a barkeeper.

1 Introduction

One of the most challenging areas of robotics during the recent years was and still is *service robotics.* In general service robots perform no repetitive tasks, e.g. guidance of a robot arm using tactile interfaces or pick and place objects in man-made environments with people around. Because these robots work together with humans in the same work space their first priority is to guarantee the human's safety. Each action has to be planned on-line in a dynamic and partially known environment. Powerful motion control structures are demanded to execute motion plans and avoid collisions.

In most cases motion planners are designed to solve n-dimensional problems in general. For planning in spaces of dimension greater than 4 commonly probabilistic algorithms are used [1, 2, 3]. The application of these planners to robotics is often straight forward manner, i.e. task specific constraints, kinematically characteristics or environmental features are usually not considered. Some algorithms take dynamical constraints under consideration [4, 5]. In general these planners take seconds to minutes to solve a problem. When the environment changes, the planners have to re-plan a collision-free path. Since planning needs a long time, recent work was focussed on combining planning techniques with fast reactive control schemes [6, 7]. A computed path need not necessarily be re-planned by the planner even if the environment changes. The reactive control structure modifies the path in a way that collisions do not occur. [8, 9] proposed frameworks where reactive control schemes are merged with planning components.

The man-machine interaction scenario in dynamic environments with moderate complexity requires a fast responding robot. Therefore this work proposes an integrated control architecture: A computationally simple motion planner interacts with reactive modules avoiding obstacles and negotiating collisions. Task oriented motions are executed according reactive plans biased by a target configuration. In determining reactive plans characteristics of the robot kinematics are considered, i.e. similar to humans either left–handed or right–handed (pick or place) plans are generated.

Section 2 to Section 4 describe the architecture and its components. In Section 5 experimental results of a barkeeper scenario are presented. A conclusion is given in Section 6.

2 Architecture for Reactive Motion Planning

To fit the requirement of a fast responding robot control this work proposes a hybrid control architec-

---

[*] This work was partially funded by the German Federal Ministery of Education and Research (BMBF) under contract no. 01IL902D0 (MORPHA).

ture combining planning and reactive components, see Figure 1.

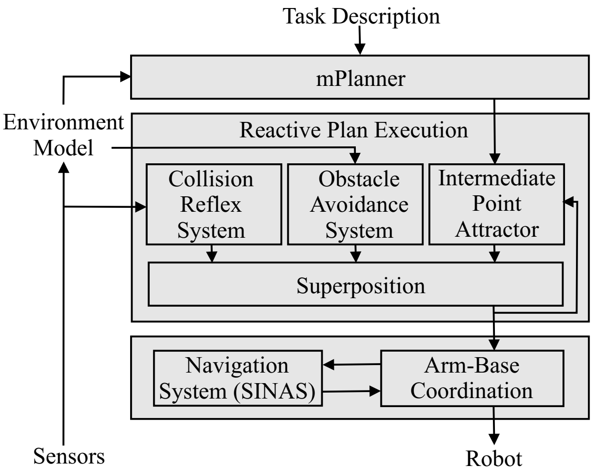

Figure 1: Architecture for *reactive* motion planning

The *mPlanner*[1] generates not the whole trajectory, but just collision free intermediate configurations (arm positions in joint space) in compliance with a given task description. Details of the planner are described in Section 3. The *Reactive Plan Execution* uses these configurations to guide the robot reactively to a desired target configuration. Several concurrent modules contribute to the computation of speed commands for arm joints and the base, see Section 4.1.

Figure 2: Experimental system

Particularly for large-scale motions of the robot platform we use our *Navigation System (SINAS)* [10]. During manipulation tasks normally the platform is placed in front of the target work space. If necessary the *Reactive Plan Execution* generates speed commands for the platform to support the arm motion, e.g. sensor-guided door opening [11] or large-scale workspace motions commanded by tactile input from a human operator [12]. The *Navigation System* is integrated by the *Arm-Base Coordination*.

Our approach was implemented on a mobile manipulator. Figure 2 shows our 8 degree of freedom (DOF) manipulator arm mounted on a non-holonomic platform. The *environment model* is built based on the geometry of the robot itself, pre-knowledge of the environment and sensor information of a stereo camera system and a laser scanner mounted on top of the robot. A tactile skin provides sensor signals for the *Collision Reflex System* whenever the arm is in collision with an obstacle. The tactile sensor also subscribes information about the surrounding of the robot for the environment model. [11] gives a general description of our system.

3 mPlanner

Purely reactive systems tend to be trapped in local minima. In our approach the *mPlanner* generates appropriate intermediate configurations to bypass local minima. For this purpose we assume that typical working environments can be structured into a free region around the robot platform and cluttered regions reaching to the start and end points of the robot motion, as illustrated in Figure 3. The algorithm is not complete and fails in high complex scenes that are rare in an everyday environment. For these rare situations where the *mPlanner* fails a probabilistic planner as listed in 1 will be taken, to determine a collision-free path. So far a probabilistic planner was not required

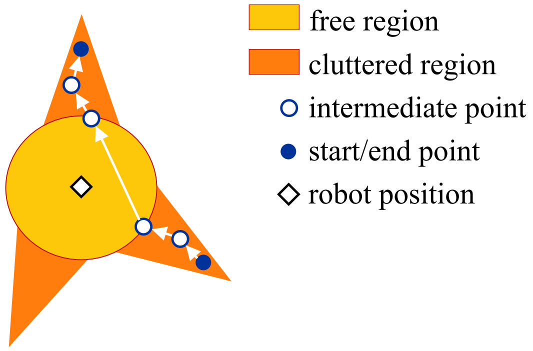

Figure 3: Supposition of a typical workspace

---

[1] *mPlanner* is the acronym for *mini*Planner. It is a fast planning component well-suited for everyday environments.

for our experiments. Currently only the *mPlanner* is implemented on our robot.

In the first step a robot configuration for the tool center point (TCP) of the end position, see Figure 3, is calculated by the *mPlanner*. The TCP is given as a position (x, y, z) and an orientation (rx, ry, rz). To calculate the arm configuration for the given TCP the inverse solution of the robot kinematics is used. Since our arm is redundant we use a set of possible solutions Q to find configurations that minimize the required configuration changes from start to end point. A collision free configuration $\underline{q}$ closest to the given start configuration $\underline{y}$ is chosen

$$\underline{q} = \min_{\underline{q}_i \subseteq Q} \|\underline{\underline{G}}\left(\underline{q}_i - \underline{y}\right)\| \quad . \tag{1}$$

n equals the degree of freedom and $\underline{\underline{G}}$ is a diagonal weighting matrix for the robot links. The configuration closest to a desired configuration $\underline{y}$ found by equation (1) is taken as an approximation for the nearest possible configuration. To save computationally complexity we do not consider the task of finding the nearest configuration as an optimization problem. Local minima due unfavorable start configurations are avoided by using an explicit set of possible configurations.

In the next step the *mPlanner* determines intermediate configurations within the cluttered regions containing the start and end point. For both regions a heuristic determines intermediate configurations that try to use as much as possible space of the arm that is used at the start and end configuration. This heuristic uses the fact that the space occupied by the robot is free of obstacles. The space additionally required by motion through the cluttered region is reduced by choosing intermediate points along the arm. Obstacles and collisions occurring in the additional space required during motion are handled by the *Reactive Plan Execution*.

4 Reactive Plan Execution

Motion plans of the *mPlanner* are transformed into joint speed commands by the *Intermediate Point Attractor* and superimposed with the joint speed commands of the *Collision Reflex System* and the *Obstacle Avoidance System*, see Figure 1. This section describes these systems in detail.

4.1 Intermediate Point Attractor

The *Intermediate Point Attractor* takes the sequence of intermediate configurations generated by the *mPlanner* switching to the next configuration when reached the current intermediate configuration. An intermediate configuration $\underline{q}_t$ is reached when the current configuration $\underline{q}_c$ gets closer componentwise than a defined threshold $\underline{a}$.

$$q_c^j - q_t^j < a^j \quad . \tag{2}$$

The threshold for the end configuration is smaller than for intermediate configurations to get a higher position accuracy of the robot.

The determined joint speeds for an intermediate configuration move the robot from a current configuration to the next intermediate configuration in minimal time. Therefore a maximum joint velocity $\underline{\dot{q}}_{\text{int}}$ under consideration of the joint acceleration limits $\underline{\ddot{q}}_{\max}$ is determined componentwise:

$$\dot{q}^j_{\text{int}} = \sqrt{2(q_c^j - q_t^j)\ddot{q}^j_{\max}} - 2t_s\ddot{q}^j_{\max} \tag{3}$$

The direction of motion is given by

$$e^j = \operatorname{sign}(q_t^j - q_c^j) \quad . \tag{4}$$

Because equation (3) demands a constant cycle time t_s we switch to a proportional controller, if we are closer to the intermediate configuration than

$$q^j_\Delta \geq \frac{4t_s^2\ddot{q}^j_{\max}}{2} \quad . \tag{5}$$

The resulting control law can be written as:

$$\dot{q}^j = \begin{cases} e^j \cdot \min\left(\dot{q}^j_{\text{int}}, \dot{q}^j_{\max}\right) & |q_c^j - q_t^j| > q^j_\Delta \\ k(q_c^j - q_t^j) & |q_c^j - q_t^j| \leq q^j_\Delta \end{cases} \tag{6}$$

For a smooth switch between both control laws in equation (6) the factor of the proportional controller is chosen by

$$k = \frac{\sqrt{2q^j_\Delta\ddot{q}^j_{\max}} - 2t_s\ddot{q}^j_{\max}}{q^j_\Delta} \quad . \tag{7}$$

4.2 Obstacle Avoidance System

The collision avoidance [13] is purely reactive. Based on an environment model repulsive virtual forces are computed.

For the collision avoidance strategy the robot is segmented into segments S_1 through S_9 according to the kinematic structure, see Figure 4. A safety distance $d_{\min}$ is assigned to each robot segment. The size of the safety distance depends on the progress of the current trajectory. E.g. at the end point, when we grasp an object, the minimum distance between the TCP of the robot and the target object has to be

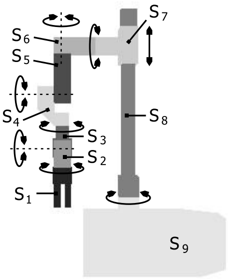

Figure 4: Robot segmentation

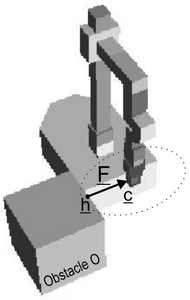

Figure 5: Virtual repulsive force generation

reduced. For each obstacle closer to a robot segment than its safety distance a virtual repulsive force

$$\underline{F} = \begin{cases} -k_{\text{obstacle}} \left[\frac{(d_{\min}-d)^2}{d}\right] \frac{\underline{h}-\underline{c}}{d} & d \leq d_{\min} \\ 0 & d > d_{\min} \end{cases} \tag{8}$$

applied to the robot segment is determined, see Figure 5. The distance d describes the distance of an obstacle to a robot segment. A minimum distance $d_{\min}$ is defined for each segment. The points $\underline{h}$ on the obstacle and $\underline{c}$ on the segment establish the minimum distance between the obstacle and the segment of the robot. Considering the control point $\underline{c}$ a torque

$$\underline{M} = \underline{F} \times \underline{c} \tag{9}$$

can be computed.

The force (8) and torque (9) vectors are combined in a force/torque vector

$$\underline{\delta} = \begin{pmatrix} \underline{F} \\ \underline{M} \end{pmatrix} \tag{10}$$

and transformed into joint space by the transposed Jacobian [14] resulting into joint torques

$$\underline{\tau} = \underline{\underline{J}}^T \underline{\delta} \ . \tag{11}$$

Using the weighting matrix $\underline{\underline{M}}$ the joint velocities for the collision avoidance are determined by

$$\underline{\dot{q}}_{\text{coll}} = \underline{\underline{M}} \ \underline{\tau} \quad . \tag{12}$$

4.3 Collision Reflex System

The *Collision Reflex System* handles collisions detected by the tactile skin which is mounted on our robot arm. Collisions can occur due an incomplete knowledge of the environment or a human interacting with the robot by touching it [12].

As in the previous section the arm is segmented according to its kinematic structure, see Figure 4. On each segment several tactile sensors are mounted. If one or more contacts occur between the robot and the obstacles in the environment the sensor locates the contact positions and measures the contact pressures. Force vectors

$$\underline{F} = -k_{\text{collision}} \ p \ \underline{e} \tag{13}$$

applied to the robot segment can be determined. p describes the amount of pressure and $\underline{e}$ describes the direction of the force.

Force vectors for all collisions are combined and result into joint speeds $\underline{\dot{q}}_{\text{tact}}$ the same way as within the *Obstacle Avoidance System*, see equations (10) to (12) in Section 4.2.

5 Experimental Results

We have tested the proposed reactive motion planning architecture on our experimental system. In a barkeeper scenario[2] the robot turns into a barkeeper. Using the robots vision system, the robot can detect several kinds of drinks, e.g. orange or apple juice. A human can interact with the robot via natural speech recognition and tactile interfaces. The speech recognition is used to order and the tactile sensors are mainly used to guide the robot.

Figure 6 shows a sequence where the robot picks a drink from the magazine (lower table) and places it onto the bar (higher table). In Figure 6a the robot is in a start position waiting for customer commands. According to the supposed work space as illustrated in Figure 3 the robot is in free space at that position. After commanding the robot to pick up a certain kind of drink from the magazine, the robot moves towards the magazine from free space into the cluttered space. If the target object is grasped (see Figure 6c), the robot moves back into the free space (see Figure 6e). In Figure 6f to Figure 6h the robot places the grasped object onto the bar and moves back into free space again, waiting for the next user commands. During robot movement the *Obstacle Avoidance System* avoids obstacles that are known by the environment model. The motion of a task behavior can be affected by the user touching the robot. The *Collision Reflex System* generates additional motions to evade a human touch. E.g. a human can work in cooperation with the robot at the bar.

---

[2] For a detailed description of the integrated systems used in the *barkeeper* example scenario see [15].

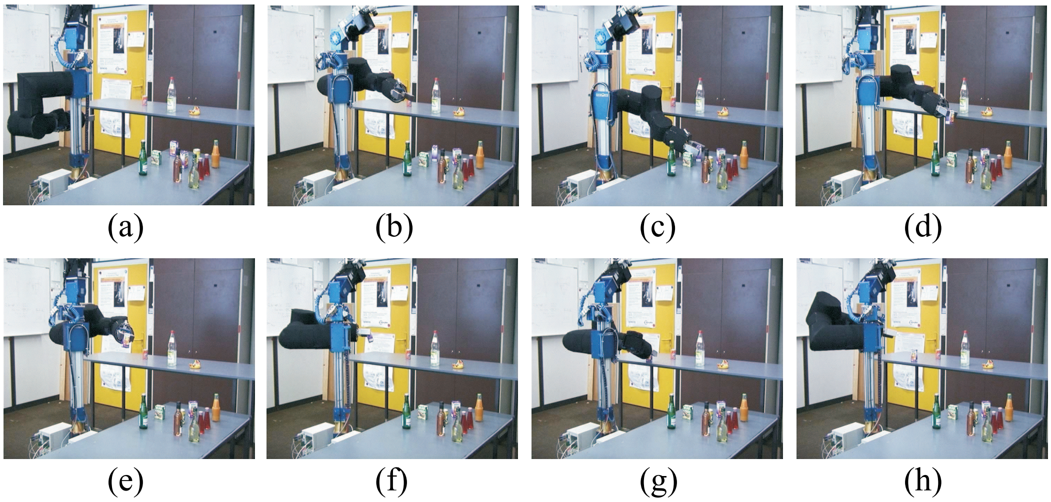

Figure 6: Robot barkeeper

6 Conclusion

This work proposes a *reactive* motion planning architecture. Specially designed to achieve a fast-responding robot control the architecture is well-suited in man-machine interaction scenarios. A simplified motion planner cooperates with fast reactive controllers. Based on an environment model obstacles are avoided during robot motion. Collisions due an incomplete model of the environment or enforced by a human are detected *via* tactile interfaces and handled in a way, the robot evades the collision. Experimental results introduce our robot as a bar keeper. Beside and during the execution of customer orders humans can freely interact with the robot, affecting the robots motion by touching and guiding the robot.

References

[1] L. E. Kavraki, P. Svetka, J. C. Latombe, and M. Overmars, “Probabilistic roadmaps for path planning in high dimensional configuration spaces”, in *IEEE Transactions on Robotics and Automation 12(4)*, pp. 566–580, 1996.

[2] J. J. Kuffner and S. M. LaValle, “RRT-Connect: An Efficient Approach to Single-Query Path Planning”, in *International Conference on Robotics and Automation*, vol. 2, (San Francisco, USA), pp. 995–1001, April 2000.

[3] A. Ladd and L. E. Kavraki, “Generalizing the Analysis of PRM”, in *International Conference on Robotics and Automation*, (Washington D.C., USA), pp. 2120–2125, May 2002.

[4] S. M. LaValle and J. J. Kuffner, “Randomized kinodynamic planning”, in *International Conference on Robotics and Automation*, vol. 1, (Detroit, USA), pp. 473–479, May 1999.

[5] R. Kindel, D. Hsu, J.-C. Latombe, and S. Rock, “Kinodynamic Motion Planning Amidst Moving Obstacles”, in *International Conference on Robotics and Automation*, vol. 1, (San Francisco, USA), pp. 537–543, April 2000.

[6] O. Khatib, “Real-time Obstacle Avoidance for Manipulators and Mobile Robots”, *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[7] H. P. Xie, R. V. Patel, S. Kalaycioglu, and H. Asmer, “Real–Time Collision Avoidance for a Redundant Manipulator in an Unstructured Environment”, in *International Conference on Intelligent Robots and Systems*, vol. 3, (Victoria, Canada), pp. 1925–1930, October 1998.

[8] O. Khatib, S. Quinlan, and D. Williams, “Robot Planning and Control”, *Robotics and Autonomous Systems*, vol. 21, pp. 249–261, September 1997.

[9] O. Brock and L. E. Kavraki, “Decomposition-based Motion Planning: A Framework for Real-time Motion Planning in High-dimensional Configuration Spaces”, in *International Conference on Robotics and Automation*, vol. 2, (Seoul, South Korea), pp. 1469–1474, May 2001.

[10] G. Lawitzky, “A Navigation System for Service Robots - From Research to Products”, in *FSR 2001* (A. Halme, R. Chatila, and E. Prassler, eds.), pp. 15–20, Yleisjäljennös–Painopörssi, 2001.

[11] G. v. Wichert, T. Wösch, S. Gutmann, and G. Lawitzky, “Mobman – Ein mobiler Manipulator für Alltagsumgebungen”, in *Autonome Mobile Systeme (AMS’00)* (R. Dillmann, H. Wörn, and M. v. Ehr, eds.), Informatik aktuell, pp. 55–62, Springer Verlag, Heidelberg, 2000.

[12] T. Wösch and W. Feiten, “Human–Robot Interaction via Tactile Interface”, in *accepted at International Symposium on Robotics and Automation*, (Toluca, Mexico), September 2002.

[13] T. Wösch and W. Neubauer, “Collision Avoidance and Handling for a Mobile Manipulator”, in *7th International Conference on Intelligent Autonomous Systems*, (Marina del Rey, USA), pp. 388–391, March 2002.

[14] M. W. Spong and M. Vidyasagar, *Robot Dynamics and Control.* Wiley, New York, NY, 1989.

[15] G. v. Wichert, C. Klimowicz, W. Neubauer, T. Wösch, G. Lawitzky, R. Caspari, H.-J. Heger, P. Witschel, U. Handmann, and M. Rinne, “The Robotic Bar – An Integrated Demonstration of Man-Robot Interaction in a Service Scenario”, in *accepted at 11th IEEE Intern. Workshop on Robot and Human Communication (ROMAN’02)*, (Berlin, Germany), September 2002.