Hierarchical Multi-robot Coordination

Viktor Seib, David Gossow, Sebastian Vetter, and Dietrich Paulus

Active Vision Group
University of Koblenz-Landau
Universitätsstr. 1
56070 Koblenz, Germany
vseib@uni-koblenz.de
http://www.uni-koblenz.de/~agas

Abstract. The complexity and variety of household chores creates conflicting demands on the technical design of domestic robots. One solution for this problem is the coordination of several specialized robots based on the master-slave principle. One robot acts as a master system, tracking and remotely controlling the slave robots. This way, only the master robot needs to be equipped with sophisticated sensors and computing hardware. We implemented a tracking system using an infra-red camera for the master and active markers on the slave robot. The master system is able to interact with the user using natural language. It builds a map of its environment automatically using a laser range finder. It can track a cleaning robot for which we use the commercially available platform "Roomba" by iRobot. The master safely navigates it to a given destination, avoiding obstacles. We successfully demonstrated the system during the RoboCup@Home competitions 2009 in Graz, Austria. We evaluate the performance of the two systems and describe the accuracy of localization and navigation.

1 Introduction

The RoboCup@Home competition concentrates on natural human-robot-interaction in typical domestic environments. In order to assist humans in accomplishing their daily tasks, robots have to be fully autonomous. That even applies in limited domains such as a household. Autonomy not only allows service robots to complete tasks on their own but also raises confidence in the machine and acceptance by human users.

The corner stones of autonomy, self localization and mapping, path planning, object recognition and manipulation with a robot arm have become an inherent part of the RoboCup@Home league. The hardware required for these purposes can be integrated on one single robot without facing major difficulties. However, even for the manipulation of objects, restricting decisions on the hardware design have to be made. Depending on the height the robot arm is mounted at, its reach is limited to certain heights. A vertically adjustable robot arm has been presented at the RoboCup@Home [SGK^{+}09]. In reserving lots of space for the

J. Ruiz-del-Solar, E. Chown, and P.G. Ploeger (Eds.): RoboCup 2010, LNAI 6556, pp. 314–323, 2011.

vertical movements this approach asks for great compromises in the arrangement of the remaining hardware.

Our approach aims at transferring certain service functions from the main system to auxiliary systems - in this case a slave robot. This way, the complexity of the main robot can be reduced. The slave robot neither needs computing hardware nor sophisticated sensors that exceed its particular task.

One can expect many future homes to be equipped with simple cleaning robots by the time that sophisticated service robots become commercially available. We present an approach to integrate such a simple cleaning robot into a household robot system by applying several extensions. The slave robot is tracked using a commercially available, economically priced infra-red camera. The tracking is carried out by the main robot in real time, which itself is a mobile platform. As slave robot we use a Roomba vacuum cleaner to perform targeted cleaning jobs. Its small size enables it to clean corners and even to reach narrow parts of a room – places that are out of range of the main robot. The modifications allow the user to instruct the master robot with tasks expressed in natural language, e.g. “clean the kitchen”. This job can be performed completely autonomously even if the vacuum cleaner is somewhere else. The main robot locates it and navigates it to the desired location. After the task is completed the slave robot is navigated back to its initial position. A benefit arising from our approach is the ability of the main robot to monitor the areas already cleaned by the slave, thus enabling the slave robot to perform a complete coverage of the cleaning area.

The approach presented here can be easily transferred to other robots to cooperate in a household environment.

The next section describes previous works dealing with the coordination of multiple robots. In chapter 3 our approach is presented in detail, the results are described in section 4. Finally, chapter 5 deals with possible future extensions of our approach.

2 Related Work

In this chapter we describe previous work on fields related to the coordination of multiple robots and point out differences to our approach. Apart from multi-robot coordination, division of tasks among robots and master-slave systems, we also present related work on cleaning robots and robot tracking due to the character of the slave robot we use.

2.1 Multi-robot Coordination

The idea of using several robots to complete one task is widely studied. In [How06] an approach for multi-robot SLAM is presented. To complete this task, multiple robots can be placed at (unknown) random initial positions. Whenever two robots meet for the first time, they use their previously recorded measurements to fuse their mapping data into one map. This approach works in real time.

Another example of solving a complex task using multiple robots is presented in [BMF+00]. The individual robots spread to different target positions for fast exploration of an area. This probabilistic approach takes into account the utility of the target position as well as the costs of reaching it. The utility is determined by the unexplored area a robot can cover when reaching it. Whenever a robot chooses a position, its utility for other robots decreases.

The soccer playing robots in the RoboCup also have to coordinate themselves. [VV03] presents an approach of coordination and task assignment based on shared potential fields. Although this simple approach is not scalable to large numbers of robots, it can be successfully applied to small teams.

Another example for multi-robot coordination is [LAL+04]. It presents a distributed architecture for multi-robot task allocation based on an enhanced Contract-Net protocol.

In all of these examples, the robots have identical capabilities and are organized without using a hierarchy.

2.2 Division of Tasks

In the RoboCup@Home finals 2009, team NimbRo simultaneously used two robots, which were designed for different tasks. The strength of "Robotinho", the first robot, was communication - both verbally and mimically [FBE+09]. Whereas the other robot, "Dynamaid", handled object manipulation at variable heights. It picked up items from tables and from the floor. Robotinho walked the guest around the apartment, Dynamaid served a drink [SGK+09]. Due to the fact that these two actions took place in parallel, they were independent. An interaction between the two robots has not taken place. Instead they interacted simultaneous with a human. This is also a demonstration of transferring services to a second robot, although the robots here were – unlike in our approach – at an equal hierarchy level and both complex systems.

2.3 Master-Slave Systems

Robotic master-slave systems are used in minimally invasive surgery, where instruments are inserted into the human body to perform medical procedures [YTEM02], [TPM03]. The benefits are improved precision and less pain to the patient. Such an instrument (slave) is controlled through an interface (master) by a human operator. The works in the medical field deal with increasing precision and reliability of these systems.

2.4 Cleaning Robots

One main goal of previous works on cleaning robots is to achieve a complete coverage of the cleaning area [NdCVVR97], [SCPZ04]. These two works use different map representations and planning algorithms to complete this task. In [JN02], a dynamic approach for multiple cleaning robots is described. The whole area is divided into polygons, which can be allocated by the robots in order to be cleaned. Since allocations occur at runtime, it is possible to compensate for the breakdown of a robot, because it would not allocate further polygons. However, the polygons already allocated by the broken robot remain uncleaned. A complete coverage of the cleaning area can also be achieved with our approach, but there is no compensation for a broken robot.

[Kur06] presents a collection of possible extensions for Roomba. Some extensions can be incorporated into domestic projects, as we did with the wireless communication. In [TD07] Roomba's abilities are evaluated for research and education purposes. The authors give a detailed technical description of Roomba. They also present basic mapping and SLAM algorithms based on the assumption of only parallel or perpendicular walls. They use Roomba's odometry and bump sensors to obtain data that is then processed externally.

2.5 Tracking

The library ARToolkit[1] can be used for object tracking with the aid of a passive marker. However, this approach is not suitable for our purpose. Due to the needed size of the passive markers, utilization of ARToolkit is reasonable at short distances only. According to the ARToolkit documentation it would require a marker the size of Roomba itself to detect it from a distance of two meters.

An approach for tracking a robot by means of an active marker is presented in [CTF05]. The active marker used consists of infra-red LEDs. In contrast to our approach, the tracking cameras are located on fixed positions, so their positions are constantly known. The cameras used operate in the visible light spectrum, which complicates the procedure of detecting the marker in the camera image. Our approach bypasses this step by the application of a specialized infra-red camera. Although specialized, the camera we use is a low-cost commercial product.

3 Our Approach

We have implemented an autonomous hierarchical multi-robot system for household chores. The main robot localizes itself and the slave robot. Note, that the localization and tracking of the slave robot is achieved in real time from a mobile platform. To track the slave, the master uses an infra-red (IR) camera. It allows it to detect the active marker consisting of IR LEDs, which is placed on top of the slave. In this scenario the master robot remotely controls the slave robot Roomba, a commercial vacuum cleaning robot built and distributed by iRobot[2]. By determining its own position and the position of the slave, the master navigates the cleaning robot safely to the desired location. Using the slave's capabilities the master fulfils a task previously given by a human user. Since the slave robot gets all required information from the master, it has no need for expensive sensors. This allows for concentrating a major part of the hardware expenses on the master robot. Furthermore, by transferring tasks to an auxiliary system, some of them can be executed more efficiently. In our example, the cleaning robot reaches even narrow parts and corners of a room.

[1] www.hitl.washington.edu/artoolkit
[2] www.irobot.com

3.1 Hardware Design

Our mobile platform "Lisa" has already participated successfully in RoboCup@Home in 2008 and 2009. In this work, it takes the role of a master robot. Lisa localizes itself using a laser range finder. A pan-tilt unit (PTU) is mounted on top of it. It can be rotated horizontally by 180° in each direction. Vertically, its possible positions range from 30° upwards to 80° downwards. We are using the IR camera of a Nintendo Wiimote mounted on the PTU to localize the slave robot.

The role of the slave robot is taken by Roomba [TD07]. Several ways to expand Roomba's abilities are presented in [Kur06]. The wireless connection to the slave robot as implemented here, was inspired by this book. We placed an ASUS WL500G Premium router on the slave. Thus, the master can send commands by opening a network socket to the router on top of Roomba. One of the USB ports of the router is used to communicate with the robot. Unfortunately, the original router firmware does not allow arbitrary data transfer through the USB ports. Instead, we had to use a modified firmware, called dd-wrt[3], a mini Linux system. To communicate with Roomba, we use the Roomba-Open-Interface protocol specification provided by iRobot. This open protocol allows to send control commands to Roomba via a serial connection. The hardware design is illustrated in Fig. 1.

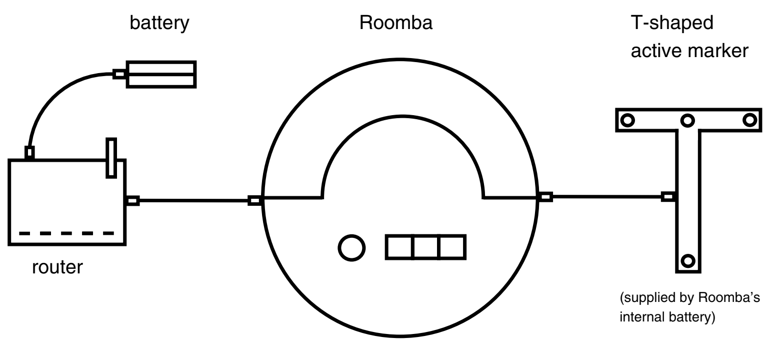

Fig. 1. Design of the hardware on the slave robot

3.2 Tracking

The projects of Johnny Chung Lee[4] inspired us to use a Wiimote, a game controller for the Nintendo Wii, for tracking. To communicate with the Wiimote we use the open library cwiid[5]. This controller includes an infra-red camera and circuits for elementary image processing. It tracks up to four infra-red sources simultaneously.

[3] www.dd-wrt.com
[4] johnnylee.net/projects/wii
[5] abstrakraft.org/cwiid/

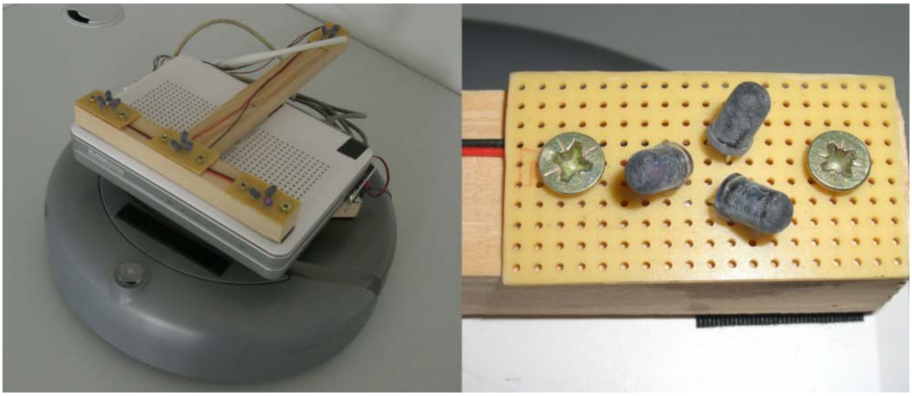

Fig. 2. Modified Roomba (left), One out of four infra-red LED groups (right)

This is sufficient to determine the position and orientation of one slave robot. If more than one slave robot is to be used, each one has to carry a marker with a different arrangement of IR sources. In order to distinguish these markers and thus the slave robots, it is necessary to use a different IR camera. In that case the IR camera's image has to pass more complex computations in order to find and identify the different markers.

We place an active T-shaped marker with IR-LEDs on top of the slave robot. The IR-LEDs that are used in the Wii sensor bar can be tracked from a distance of five meters if pointed vertically at the Wiimote. Since we expect to point at the slave robot's infra-red marker at angles between 25 to 55 degrees to the vertical (i.e. the PTU is tilted 35 to 65 degrees), the light sources have to be slightly modified. To increase the maximum tracking distance, the four infra-red sources are composed of three LEDs. Each one is tilted outwards at an angle of 40 degrees (see Fig. 2) thus emitting the IR light into the expected direction of the IR camera.

As it is only sensitive to IR light, the infra-red camera allows to bypass the task of (possibly erroneous) image processing in order to find the slave robot: the only thing contained in the image is the active marker. Hence computation time can be saved. With the detected marker the relative position of the slave to the main robot can be calculated. To accomplish this, we use a graph-based representation of the robot geometry. It is permanently updated to reflect the movements of actuators or the robot itself (see Fig. 3). As the IR camera is attached to the PTU, its position and orientation is always known in local robot coordinates.

The IR camera's sensor[6] yields 2D coordinates of the detected infra-red sources. Depth information cannot be obtained in this manner. Therefore we face the challenge of converting 2D sensor coordinates to 3D local robot coordinates.

[6] Note that we ignore the radial distortion of the camera in the following transformations.

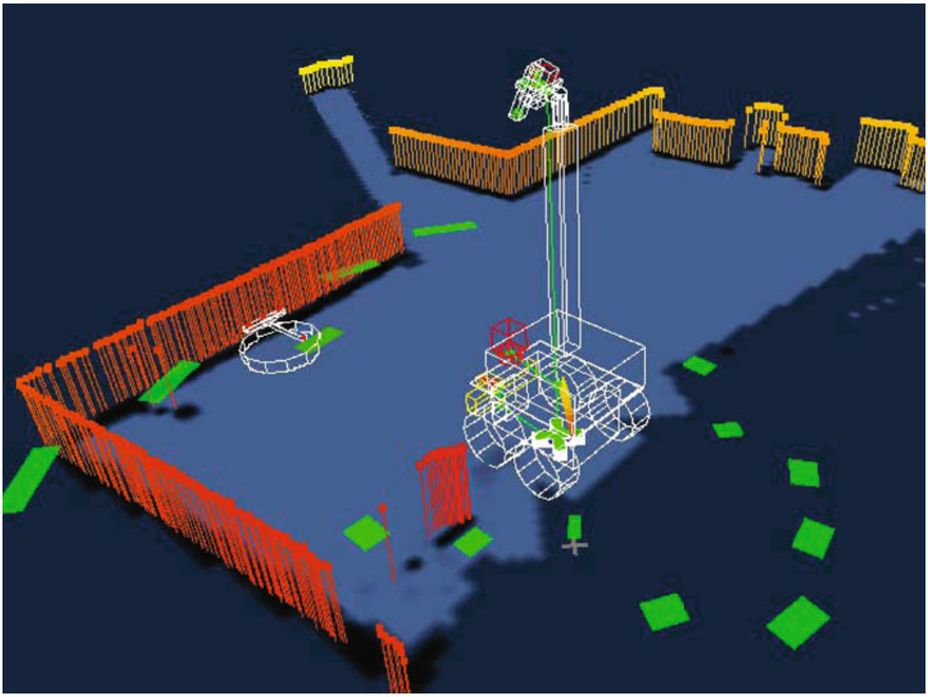

Fig. 3. The main robot's belief about its state and the slave's pose

First we need to determine the distance of the IR camera to the IR sources before applying the scene graph's transformation. This is done as follows: Let x and y be the 2D coordinates of a detected IR source. Furthermore, $w = 1024$ and $h = 768$ describe the resolution of the camera's sensor. Then we apply the following transformation to the 2D coordinates:

$$p_x = \frac{2 \cdot x}{w} - 1 \qquad p_y = \frac{2 \cdot y}{h} - 1 \tag{1}$$

As no depth information is available, we assign an arbitrary value unequal to zero to p_z. Using the scene graph, this point $\boldsymbol{p}$ is transformed into local coordinates. Now the transformed coordinates are located somewhere on a straight line through the IR camera and the IR LEDs - depending on the value that was assigned to p_z. Since the slave robot always moves on the ground, we can now determine its position. We calculate the vector $\boldsymbol{v}$ from the camera's position $\boldsymbol{c}$ through point $\boldsymbol{p}$ and intersect the corresponding line with the plane the active marker is in, which is parallel to the ground plane. Let the height of this plane be h_{led}. Then the line parameter r can be determined as

$$r = \frac{h_{led} - c_z}{v_z} \tag{2}$$

where $v_z = c_z - p_z$. Thus, the position of the IR light source in local coordinates is

$$\begin{pmatrix} x_r \\ y_r \\ z_r \end{pmatrix} = \begin{pmatrix} c_x + r \cdot v_x \\ c_y + r \cdot v_y \\ h_{led} \end{pmatrix} \quad (3)$$

Since $\boldsymbol{v}$ always needs to intersect with the ground plane, this method works only if the IR camera is tilted towards the ground. Whenever the computation yields an upwards directed vector or a point further away than $2.5m$, we have detected an erroneous infra-red source (for example a sun reflection). In this case the point is ignored.

In this manner all of the four IR light sources on the active marker are converted. Now we make use of the T-shaped marker to determine the slave's orientation and obtain its pose in local robot coordinates of the main robot.

3.3 SLAM

Based on laser scans and odometry information, the main robot continuously generates a grid map of its environment and localizes itself in it. It is based on the Hyper Particle Filter (HPF) concept [PP09]. The HPF contains a fixed number of particle filters running in parallel, each one with its own map and distribution of elementary particles. Each elementary particle contains information about the robots position and orientation. For each particle filter, some measurements are randomly ignored, so that the results vary. The maps generated by each particle filter are weighted according to their contrast, which is an indicator of quality.

3.4 Path Finding and Coordination during Navigation

Using the knowledge of the master's position, the global position of the slave robot can be determined, thus providing a basis for path planning. The path planning algorithm is described in [WP07].

The slave robot always moves in two steps. First, it aligns to the next waypoint. Then, it moves to a certain distance in the new direction. After it has finished its movement, the slave's position is determined again so that it can move to the following waypoint. During the navigation phase, the master robot constantly adjusts the IR camera to the slave's position. If the slave moves too far away or the intervisibility is lost, the master sends a stop command. In this case it approaches the slave's last known position and tries to find the slave by systematically tilting and panning its PTU.

4 Results

To evaluate the infra-red marker we use the program wmgui[7]. It provides a graphical user interface in which it displays the detected infra-red sources by the Wiimote and their intensities.

If pointed vertically at a LED group the Wiimote detects the infra-red light up to a distance of $3.5m$. The minimal distance is $30cm$.The whole active marker

[7] packages.debian.org/de/sid/wmgui

is detected up to an angle of 70 degrees to the vertical in a maximum distance of two meters. At higher angles or in greater distances not all of the four light sources are reliably detected. When in operation, the IR camera is pointed at the marker in angles between 55 and 25 degrees to the vertical, which corresponds to PTU tilt angles of 35 to 65 degrees. The distance to the slave robot is limited to two meters, whereas the angle does not restrict the operation.

Depending on the angle and orientation the slave is located in, some of its IR LEDs can be detected up to distances of three meters. Since not all of the four IR sources can be detected at this distance, a safe navigation and precise pose estimation is not possible.

The master is able to find the slave on its own in short time, if the intervisibility is lost. This can be achieved due to an immediate stop command, that is send to the slave and the systematic search using the PTU. However, when the slave is taken away and put on the ground further than two meters away from the master, it loses its position and cannot continue operation.

When the cleaning robot moves under a table or a comparable obstacle, the intervisibility is lost and cannot be recovered. Furthermore, this approach is limited to even floors, since the pose estimation of the slave robot relies on this fact.

5 Conclusion

We successfully implemented the interaction of two robots based on the master-slave principle. The master can reliably track the slave robot up to a distance of two meters and navigate it to a target in arbitrary distance. It operates indoors on even floors and relies on the intervisibility between the robots.

To avoid the necessity of permanent intervisibility, the slave robot's odometry data could be used to compensate for short-time loss of intervisibility. In our scenario, this extension would allow the cleaning of occluded areas, e.g. under a table or bed. The tracking system could also be extended by cameras installed at fixed positions in the scenario. After guiding the slave to a monitored area, the master robot could pursue other tasks while tracking the slave via the fixed cameras.

Although we use specific robots for our experiments, the approach described in this paper can be seen as a general solution for integrating the current generation of commercial household robots with a higher-level autonomous system.

References

[BMF+00] Burgard, W., Moors, M., Fox, D., Simmons, R., Thrun, S.: Collaborative multi-robot exploration. In: IEEE International Conference on Robotics and Automation (2000)

[CTF05] Cassinis, R., Tampalini, F., Fedrigotti, R.: Active markers for outdoor and indoor robot localization (2005)

[FBE+09] Faber, F., Bennewitz, M., Eppner, C., Görög, A., Gonsior, C., Joho, D., Schreiber, M., Behnke, S.: The humanoid museum tour guide robotinho (2009)

[How06] Howard, A.: Multi-robot simultaneous localization and mapping using particle filters (2006)

[IB98] Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. International Journal of Computer Vision 29(1), 5–28 (1998)

[JN02] Jäger, M., Nebel, B.: Dynamic decentralized area partitioning for cooperating cleaning robots (2002)

[Kur06] Kurt, T.E.: Hacking Roomba: ExtremeTech. John Wiley & Sons, Inc., New York (2006)

[LAL+04] Lemaire, T., Alami, R., Lacroix, S., LAAS, C., Toulouse, F.: A distributed tasks allocation scheme in multi-UAV context. In: Proceedings of IEEE International Conference on Robotics and Automation, ICRA 2004, vol. 4 (2004)

[NdCVVR97] Neumann de Carvalho, R., Vidal, H.A., Vierira, P., Ribeiro, M.I.: Complete coverage path planning and guidance for cleaning robots. In: IEEE Int. Symposium on Industrial Electronics (July 1997)

[Pel08] Pellenz, J.: Mapping and map scoring at the robocuprescue competition. In: Quantitative Performance Evaluation of Navigation Solutions for Mobile Robots (RSS 2008, Workshop CD) (2008)

[PGP09] Pellenz, J., Gossow, D., Paulus, D.: Robbie: A fully autonomous robot for robocup rescue. Advanced Robotics (Robotics Society of Japan) 23(9), 1159–1177 (2009)

[PP09] Pellenz, J., Paulus, D.: Stable mapping using a hyper particle filter. In: Baltes, J., Lagoudakis, M.G., Naruse, T., Ghidary, S.S. (eds.) RoboCup 2009. LNCS, vol. 5949, pp. 252–263. Springer, Heidelberg (2010)

[SCPZ04] Seop, J., Choi, Y.H., Park, J.B., Zheng, Y.F.: Complete coverage navigation of cleaning robots using triangular-cell-based map. IEEE Transactions on Industrial Electronics 51(3), 6 (2004)

[SGK+09] Stückler, J., Gräve, K., Kläß, J., Muszynski, S., Schreiber, M., Tischler, O., Waldukat, R., Behnke, S.: Dynamaid: Towards a personal robot that helps with household chores. In: Proceedings of RSS 2009 Workshop on Mobile Manipulation in Human Environments, Seattle (June 2009)

[TD07] Tribelhorn, B., Dodds, Z.: Evaluating the roomba: A low-cost, ubiquitous platform for robotics research and education (2007)

[TPM03] Tavakoli, M., Patel, R.V., Moallem, M.: A force reflective master-slave system for minimally invasive surgery. In: Proceedings of the 2003 IEEE/RSJ Int. Conference on Intelligent Robots and Systems (October 2003)

[VV03] Vail, D., Veloso, M.: Multi-robot dynamic role assignment and coordination through shared potential fields (2003)

[WP07] Wirth, S., Pellenz, J.: Exploration transform: A stable exploring algorithm for robots in rescue environments. In: Workshop on Safety, Security, and Rescue Robotics, pp. 1–5 (September 2007), `http://sied.dis.uniroma1.it/ssrr07/`

[YTEM02] Yamano, I., Takemura, K., Endo, K., Maeno, T.: Method for controlling master-slave robots using switching and elastic elements. In: Proceedings of the 2002 IEEE International Conference on Robotics & Automation (May 2002)