Exploiting Structure in Two-armed Manipulation Tasks for Humanoid Robots

Franziska Zacharias, Daniel Leidner, Florian Schmidt, Christoph Borst, and Gerd Hirzinger

***Abstract*— In autonomous bimanual operation of a robot, parallelized planning and execution of a task is essential. Elements of a task have different functional and spatial relationships. They may depend on each other and have to be executed in a specific order or they may be independent and their order can be determined freely. Consequently, individual actions can be planned and executed in parallel or not. In a proof of concept, this paper shows that the structure of a task and its mapping onto subordinate planners can significantly influence planning speed and task execution. Independent tasks are planned using two parallel path planners. Dependent tasks are planned using one path planner for both arms. Using a simple, yet expandable experimentation scenario, the resulting recommendations for parameterizing path planners are verified on a humanoid robot. For execution on the real robot a violation of the rigid body model used in path planners had to be addressed.**

I. INTRODUCTION

If a humanoid robot should serve as an assistant in the household (Fig. 1), it has to be able to analyze a task and decide how to execute it. To be accepted, a household assistant has to be able to mimic the reasoning process and the execution speed of humans. Humans use mostly the arms and the upper body for manipulation tasks. The upper body extends the workspace of each arm, while the two arms allow parallelized or cooperative task execution. In bimanual tasks, parallelized planning and execution of elements of a task is essential. For this the parameterization of low-level planners, like path planners, is a key issue. In current research, elementary one-armed or two-armed manipulation tasks involve the grasping of objects. A fact not yet addressed in the planning literature is that the structure of a task, i.e. the relations of actions among each other, can significantly influence the planner setup, planning speed and task execution.

Using a simple, yet easily expandable scenario for two-armed manipulation, this paper presents a proof of concept. The benefits of preserving and exploiting the functional characteristics of task elements are shown. Using the example of path planning, it is shown that instead of changing the internal mechanisms of a planner, the task structure can be preserved through appropriate setup of the planner. Keeping independent tasks independent on subordinate planning levels results in faster planning and execution of the task. Evaluation is performed in simulation and on the real humanoid robot system *Rollin' Justin*.

All authors are affiliated with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Wessling, Germany, franziska.zacharias@dlr.de

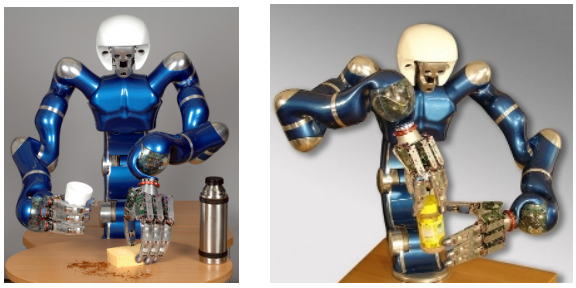


Fig. 1. Justin performing bimanual tasks.

II. PROBLEM ANALYSIS

In this section various views on a given task are considered. First, the elements of a task are analyzed on a logical level. It is shown that a task can be structured according to functional characteristics. Subsequently, the task is also analyzed on a geometrical level. Second, it is examined how state of the art planning systems map these characteristics onto low-level planners. Finally, bimanual tasks are examined and possibilities are presented to preserve the structure of the task.

A. Structure of a task

What is the advantage of bimanual task execution? With one arm and hand only a limited set of tasks is solvable. The second arm and hand often has to keep an object immobile, e.g. in unscrewing a cap. Two-armed and two-handed task execution is a fundamental feature of human manipulation processes. The human having two arms extends the workspace reachable for the human without needing torso or leg motion. Bimanual execution speeds up the task's completion. Bimanual tasks are comprised of a set of subtasks for each arm. Since each subtask can again be broken up into multiple tasks on a lower level, only the term *task* will be used from now on. Guiard [1] roughly structures tasks into unimanual tasks and symmetric or asymmetric bimanual tasks. The later always have some temporal or spatial dependencies. This paper adopts a still more diversified view on the problem considering logical and geometric connections.

On a logical level where a task planner structures the task, actions have different functional characteristics. They can be independent or they can have temporal, or functional connections. The task of setting a table for breakfast (Fig. 2) will be used to illustrate possible action categories. Placing objects such as milk bottles, fruits or a sugar bowl is best

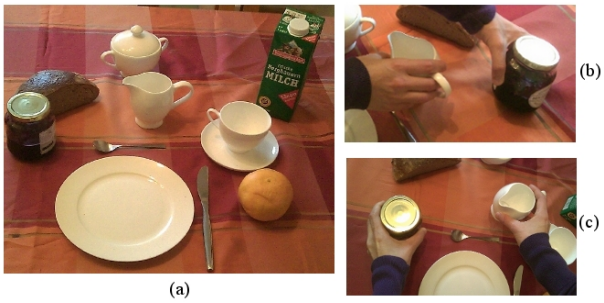


Fig. 2. Table set for breakfast (a). And logically independent actions that (b) have or (c) do not have geometric dependencies.

described by the category of one-handed pick and place tasks. Individual actions of this category are logically independent of one another. A second category can be illustrated by the task of setting a cup on a saucer. This requires to place the saucer first before the action of positioning the cup can be fulfilled. A sequential condition is involved. So far, a two-armed system is still not needed to ensure task success. However, opening a milk bottle involves fixing the position of the bottle and unscrewing the cap. This is clearly a bimanual task falling into the category of cooperative tasks. Temporal and functional interdependencies can be identified. In view of execution of a task on a humanoid robot, tasks have to be examined on the geometric level. The tasks can be characterized as spatially dependent tasks (Fig. 2 b) or spatially independent tasks (Fig. 2 c). Spatially independent tasks do not influence each other, i.e. the task goal attributed to one arm does not restrict the workspace of the other arm. Tasks are considered to be spatially dependent if the action of one arm influences that of the other arm. This may concern changes of the environment through object displacements or possible collisions between the arms during task execution. Cooperative bimanual tasks are always spatially dependent. In general, a number of possible action categories identified on the logical level of the task planner, can be further differentiated on the geometrical level. According to the outlined dependencies, the task planner could structure the order of the planning and execution steps. The next section analyzes how tasks are currently mapped to low-level planners, such as path planners which are essential for successful task completion.

B. Mapping onto subordinate planners

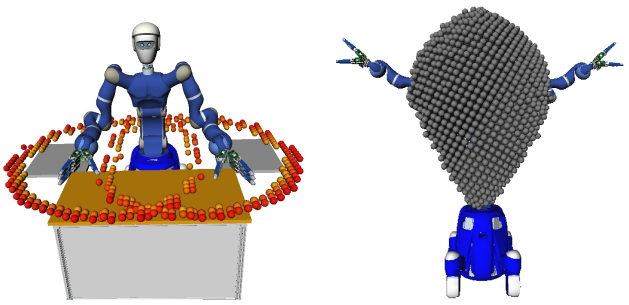
As more humanoid robots with dexterous hands like Armar [2], Justin [3], or TwendyOne [4] are developed, research has begun to focus on elementary one-armed or two-armed manipulation tasks. Vahrenkamp et al. [5] combine the inverse kinematics for a humanoid robot with RRT-based motion planning. The aim is to find whole-body configurations and trajectories suitable for grasping an object with both hands at once. Diankov et al. [6] search through work- and configuration space when solving a planning task for approaching an object and grasping it one-handed. They also exploit information about the robot's kinematic structure, e.g. base reachability to speed up planning. Vahrenkamp et al. [7] generate grasping motions for a humanoid robot. A separate planner is started for each arm. The planner that finishes first, determines the arm to be used for grasping. However only the grasping of a single object is addressed without a higher-level task context. Berenson et al. [8] present a motion planning algorithm for bi-manual interaction with objects that are subject to pose constraints. The task constraints strongly restrict the allowed subspaces for the planning making planning encompassing the whole body of the legged humanoid robot feasible. Koga et al. [9] present a two stage approach to address multi-arm manipulation planning. The arms cooperatively manipulate the same object.

A fact currently neglected in the planning literature is that the structure of the task also significantly influences the planner setup, planning speed and task execution. The only aspect that has been addressed is that a humanoid robot can be considered as a structured entity. Heuristics are used to separate walking and manipulation planning. If the robot is still far from the object to be manipulated the path planner does not need to consider the robot arms in the motion planning [10]. All introduced papers address only elementary tasks and apply the same planner throughout the task. Let us assume a humanoid robot with two arms like Justin (Fig. 1) has to solve a complex manipulation task such as setting the table. The path planning for the whole task is performed with a path planner that works in the combined configuration space of both arms. In this case, actions are coupled through path planning that have no functional connections. The structure identified on a logical level cannot be exploited anymore since the structure of the task planning problem is lost during the mapping process onto low-level planners. Thus also the benefits of decoupled, parallelized planning of independent tasks are lost.

Only papers from the programming by demonstration literature address preserving the structure of the task during the mapping onto a humanoid robot. Zöllner et al. [11] teach dual-arm manipulation tasks to a humanoid robot. The trajectories learned from the human are adapted and transferred onto the robot. However no path planners are involved and the scenarios are not extendable to arbitrary scene setups. The aim of this paper is a proof of concept. The functional structure of tasks is identified. The benefit of preserving the independence of actions on the logical level during the mapping onto subordinate planners is demonstrated. This will for instance enable parallelization and permutation of independent actions.

III. Preserving the functional structure

Recalling again the task of setting the table for breakfast, the mapping onto subordinate planners is one main issue to be solved. Assuming a two armed humanoid robot with 7 degree of freedom (DOF) arms and a state of the art path planner is available, it has to be decided how to parametrize the planner. Should one independent planner be parametrized for each 7 DOF arm or should one planner be setup for the combined 14 DOF workspace of the two arms? The



(a) (b)

Fig. 3. (a) Border of the workspace for the right and left arm on the table surface. (b) Reachable workspace volume of the right arm of Justin that is also reachable for its left arm.

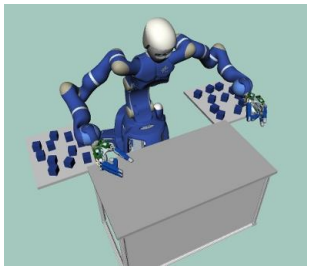


Fig. 4. Shows a start situation for a task.

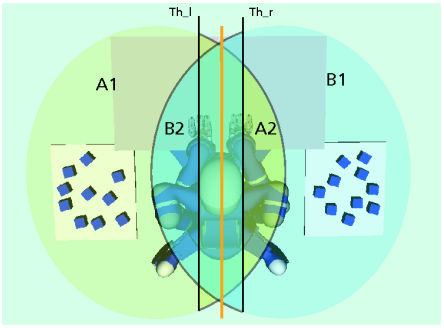


Fig. 5. Workspace of the left and the right arm on the table surface.

torso movement is not included here because the human uses it mainly if a task goal is outside an arm's reachable workspace. The scenarios for evaluating the presented ideas were chosen so, that the torso movement is not needed. First, the reachable workspace of a humanoid robot is examined. If the workspace volume of each arm is cut through at the height of the table surface, two intersecting circles result. The border of the reachable workspace of each arm at the height of the table surface is shown in Fig. 3 (a). The figure shows that the humanoid robot is able to reach all positions on the table surface and the side tables with its arms. Given the robot's base position is fixed, regions on the table can be related to one or both robot arms. In Fig. 3 (b), the volume is shown where the workspaces for the two arms overlap. This knowledge about the robot arm's reachable workspace allows labeling these regions into categories. Task elements, i.e. their goal states, either lie in disjunct parts of the workspaces for the right and the left arm or they lie in the overlapping region. The next sections will focus on the structuring of planners and the planning environment. The benefit of parametrizing path planners according to an action's functional and geometrical characteristics will be demonstrated.

IV. Scenario Setup

The goal of all scenarios is the eventual realization of a composed scenario on the real humanoid robot system. The target scenario should be arbitrarily expandable to enable the demonstration of aspects of various difficulties of bimanual task planning and execution. The grasps on the objects are assumed to be given. Simple objects are chosen to reduce the requirements with respect to object recognition and enable concentration on the planning aspects. Cubes of 0.06 m width are manipulated. In general, the task setup is composed of a number of cubes distributed in specified areas (Fig. 4). A specification of the goal state of the task is also assumed to be provided. In all experiments, the mobile humanoid robot *Rollin' Justin* [3] is used. It consists of an humanoid upper body system mounted on a mobile base with variable footprint. The humanoid upper body is composed of a 3-DOF torso, two 7-DOF arms, two 4-finger hands and a 2-DOF head. The 7-DOF DLR light weight robot arm serves as the left and the right arm of the robot *Rollin' Justin*. For path planning the open source planning architecture OpenRAVE [12] is used. Specifically, the BiRRT planner [13] implementation is used.

V. Exploiting Structure in the Task

This section discusses methods to structure the planning process to enable faster bimanual task completion. The configuration of the upper body is shown in Fig. 3 (a) and stays fixed.

A. Planning tasks in disjunct workspaces

Tasks that lie in disjunct regions of the two armed workspace, i.e. regions $A1$, $B1$ in Fig. 5, do not interfere with each other. Paths that do not collide regardless of their individual timings can be found. This can be exploited. The path planning and execution of disjunct, independent tasks can be parallelized. To accomplish this, first the planner has to be properly configured. An independent planner is setup for each robot arm. It uses all degrees of freedom of the attributed robot arm. However these considerations do not ensure that the paths, that are planned independently, are collision-free across their whole length. Randomized path planners explore the whole configuration space. Parts of the found path might intersect with the workspace of the other arm. Therefore collisions between the two arms can occur. To avoid this, a structural element is introduced into the environment. A virtual wall is added that separates the workspaces of the two arms. Virtual walls are currently used in haptic interfaces to simulate contact surfaces [14]. In this paper, it serves as a filter to discard configurations that penetrate the workspace of the other arm and thus invalidate

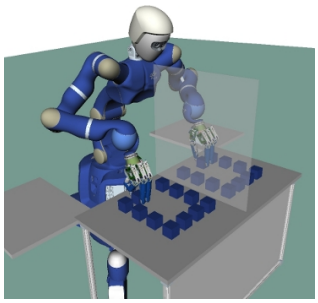


Fig. 6. Structuring the workspace for independent tasks.

time for	deadline	transit tasks	transfer tasks	total	success rate
2 x 7 DOF	60	17.4	16.7	423.4	100 %
14 DOF	60	23.6	26.2	571.6	13 %
14 DOF	120	27.5	25.4	627.5	52 %

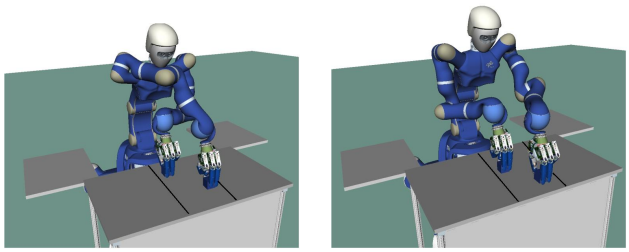
TABLE I

TIMES IN SECONDS FOR PLANNING INDEPENDENT TASKS IN DISJUNCT PARTS OF THE ROBOT ARMS WORKSPACE.

the assumption of independent actions. The virtual wall can be introduced at different levels, in the sampling step and during collision checking. In the first case, the path planner internals have to be modified. The configuration is discarded during the sampling step when the corresponding Cartesian end-effector position lies in the forbidden region of the workspace. However, the forward kinematics computations consume the time savings of the cases where the filter applied. Furthermore, this concerns only the end-effector. Other parts of the robot such as the elbow can still extend into the workspace of the other arm. Thus a virtual wall in the environment is also necessary. If an inadmissible robot arm configuration is sampled it collides with the virtual wall and is discarded. In the following evaluation the virtual wall was only added as an obstacle in the environment and the planner was not modified. To demonstrate the merits of preserving the functional independence of tasks in path planning, the planning is also performed with a planner configured to use both arms and plan in the combined configuration space of the two arms. Its performance is compared with the proposed system that exploits robot specific knowledge to structure the planning problem.

Evaluation: For evaluation a scenario was chosen that involves only independent tasks. Each arm has to place cubes on the table in the pattern of a square. Fig. 4 shows the scene at the start. 22 cubes are placed on two side tables, 11 for each arm. Fig. 6 shows the desired final setup with Justin placing the last cubes. The virtual wall is shown that ensures independent working areas. Always the cube was manipulated next that had the smallest euclidean distance to the center of the main table. The two-handed task was planned with two independent, parallel-running path planners, one for each 7 DOF arm. This system setup is labeled *2x7 DOF* system. Additionally, results are shown for one path planner that explores the arms' combined 14 DOF configuration space, labeled the *14 DOF* system. Table I shows planning times for the simulation experiment. The times include planning and optimization times. The same number of samples and optimization cycles is used for the 2x7 DOF and the 14 DOF system. In transit tasks the robot arms plan for moving empty-handed to the next cubes to be grasped. In transfer tasks the robot arms plan for transporting the grasped cubes to the designated places in the pattern. Thus two planning runs are performed per moved pair of cubes. The results were averaged over 100 runs of the whole two-handed task. The sequence of the cube placements were identical for each run. For each cube the two planners of the 2x7 DOF system are started at the same time. To determine the planning time of the 2x7 DOF system for a subtask, the time of the planner that finishes last is taken for the averaging. If the planning time for any transit or transfer task element exceeded the given deadline, the whole task was considered to have failed and was not included in the averaging. The number of failures is reflected in the success rate. A success rate of 100% means that all 100 runs of the experiment were successfully planned. In view of eleven cubes to be moved by each arm, a maximum time of 60 seconds for one run of the planner is already quite generous. As table I shows the 2x7 DOF system is always able to complete the task. Given the same deadline, the 14 DOF system that does not mirror the functional independence of the tasks is significantly slower and succeeds only in 13% of the runs. The reason for this is that in the same space of time a much smaller portion of the exponentially expanded search space is examined. Thus the probability of finding a solution in the same amount of time is significantly reduced. One would expect that the system that plans in the 14 DOF configuration space needs at least twice the time of the 2x7 DOF system. As the deadline is relaxed, it can be seen that this assumption is verified. While a deadline of 120 seconds per run of the planner is infeasible for any real execution scenario, still only a success rate of 53% is reached. The times for the transit and transfer tasks were summed up to obtain the time for completing the whole task (column *total*). Execution times are not listed as in both cases the arms move in parallel.

The results show that the two independent 7 DOF planners with the introduced virtual wall clearly outperform the 14 DOF path planner without a virtual wall. This is a clear indicator for the mapping of functional characteristics onto subordinate planners. The success rate was primarily affected by the success rate of the transit task. Here the arms moved to the next cube to be grasped. Because of other cubes near to the one to be grasped, most transit tasks have a narrow passage problem at the end. The fingers arranged in a pre-grasp shape have to move into the space between the cubes. As Tab. I shows, planning for transit tasks took longer than for transfer tasks. The 14 DOF planner ran on a computer with quad-core Intel Xeon 2.67 GHz and 3 GB main memory. The two 7 DOF planner were run in parallel on two independent computers of the type just mentioned.



(a) cross-over (b) no cross-over

Fig. 7. Task goals in the overlapping workspace regions of the two arms.

	cross-over		no cross-over	
time for	planning	execution	planning	execution
7 DOF	15.5	7.5	14.4	7.4
14 DOF	20.9	4.3	18.8	3.7

TABLE II

TIMES IN SECONDS FOR PLANNING DEPENDENT TASKS IN THE REGION WHERE THE WORKSPACES OVERLAP.

All facts combined show that a planner need not be changed to achieve better performance, only set up according to the task characteristics.

B. Planning tasks in the two-armed workspace

For tasks in the region where the workspaces of the two arms overlap, task elements may not have functional connections. However, they influence each other on the geometrical level. Again, two planner setups are possible. To mirror the geometric characteristics of the actions, a single planner is configured to plan in the combined 14 DOF workspace of the two arms. On the other hand, an independent planner is set up for each 7 DOF arm (as in Sec. V-A) and paths are planned in parallel. Since the robot arm paths might overlap to solve the task, planned paths must be checked for collisions with one another. If collisions occur the execution for each arm cannot be parallelized but must happen in a sequential manner. In this paper, the worst case is assumed and the execution is always performed in a sequential manner.

Evaluation: To evaluate the planning in the overlapping workspace of the two arms, two cubes were placed. The subregion of the overlapping workspace where the task structure cannot be preserved by a virtual wall is dependent on the size of the hands and the grasped objects. The borders of the region are denoted by the straight lines in Fig. 7. The cube positions were randomly sampled from the indicated region. Two categories of cube placements are distinguished, the case where the robotic arms cross-over (Fig. 7 a) and the case where they do not (Fig. 7 b). Table II shows planning and execution times for this experiment again averaged over 100 runs of the task. In terms of planning time, the 2x7 DOF system is faster than the 14 DOF planner. For cross-over goal positions, the differences are more pronounced. The crossed-over position of the two arms posed a narrow passage problem for the planner. One of the arms functioned like a "gate" obstacle through which the other arm had to pass. So while the environment may seem to be lacking obstacles, this is not the case. Also if the goal position of a cube is close to the upper body, the upper body generates a narrow passage problem because it strongly restricts the collision-free reachable space in the target area.

As expected, the sequential execution takes about twice the time of the parallelized execution of the tasks of the two arms. The execution time was measured in simulation by interpolating the trajectory with a resolution of 1° and a maximum link velocity of $\frac{90^\circ}{s}$. While the execution time will be different on the robot system, it was added for reference here. If planning and execution times are combined, the values for both system differ only by a second. In view of the randomized planning principle of the used path planner, this is negligible. Thus, for this simple scenario both systems perform equally well. For more differentiated conclusions, several more scenarios with an increasing number of additional obstacles have to be examined. In view of human-like two-handed manipulation, the 14 DOF planner with the parallelized execution is preferred.

VI. TRANSFER ONTO THE REAL ROBOT

The scenario to be executed on the real robotic system is composed of tasks in the disjunct and in the overlapping workspaces. As in Fig. 4, twenty-two cubes are placed on two side tables. The humanoid robot uses the above mentioned strategies to lay a given pattern, in this case the character string DLR (Fig. 9). In the disjunct workspaces, i.e. regions A1, B1 in Fig. 5, two independent planners are used, one for each 7 DOF robotic arm. In the region where the workspaces of the arms overlap a planner that plans in the 14 DOF configuration space is used to plan the task. A task is planned with the 14 DOF planner if the goal position for the right arm lies in region $B2$ to the left of line Th_r or if the goal position for the left arm lies in region $A2$ to the right of line Th_l in Fig. 5. To concentrate on the porting onto the real robot, the cube positions are assumed to be known. When going from the simulated to the real system, major and minor problems have to be dealt with. This is especially the case for robots in light weight construction because here the rigid body model of the path planner is violated.

A. Violation of the rigid body model

The robot Justin is completely based on the light weight construction principle. When an arm is e.g. stretched out, the torques acting proportionally on each link segment cause a slight deformation in the harmonic drives, more specifically in the gear structures, resulting in a slight sagging of the arm. The real position of the end-effector deviates from the position predicted by the forward kinematics.

$$\Delta q_i = \frac{\tau_i}{k_i} \tag{1}$$

The deviation Δq_i for each link i can be computed using the stiffness value k_i for each link and the torque τ_i acting on link i as shown in equation 1. Given the robot configuration, the torques τ_i can be obtained by computing the gravitation

model. Two possibilities exist for applying the corrections. On the one hand, the trajectory from the planner can be translated into appropriate motor commands that apply the computed offsets. Thus the path planner plans the trajectory on the joint-side. The advantage of this method is that the planner itself is not modified and remains valid for different types of robots. However, to avoid violating the joint limits through the offsets added later, the motion range of the joint has to be shrunk by the worst case deviation for the planning. Given the weight of the links and the hands, and assuming that standard light objects, like glasses, are grasped, the maximum deviation for each link is about 0.5 degrees. Reducing each joint range by this value is acceptable and sufficient. On the other hand, the offset can be applied in the path planning step itself. Then the planner plans the trajectory on the motor side. However, the internals of the planner have to be changed, i.e. the direct kinematics. The joint ranges are not modified. However, the direct kinematics are a central component of the planner and is computed quite often, i.e. during every sampling-step. The computations for determining the torques consume about as much processing time as a collision check. Since the computations would have to be done for each link of each arm, the time the path planner needs would increase significantly. Also the modeling will never be absolutely precise leaving a residual error to be compensated by appropriate motor commands. Therefore, the corrections for the arm links where applied after the planning step outside of the path planner.

For the fourth torso link this is not possible. The first three links of Justin's torso are active links. The fourth link is passively actuated by wire ropes. The intend was to always keep the last torso segment upright and transfer the acting torques to the base [15]. Depending on the configuration of the two arms, a torque acts on the fourth torso link. Its absolute value varies as different arm configurations are assumed. This causes the wire ropes to stretch and the 4th torso link to deviate from the position predicted by the forward kinematics. In Fig. 8 the torso is not moved. The red arrows indicate the 1.6 m mark on a yard stick. For two different configurations of the left arm, the sagging of the torso can clearly be observed by the how much the right hand sags along the yard stick. Since the fourth link is not actuated, the deviation cannot be compensated by appropriate motor commands. Therefore the direct kinematics module of the planner was modified to apply the correction defined by eq. 1. As the torso is not moved by the planner and only one link is involved, the effect on the planning speed is negligible.

B. Execution on the robot

The total planning time for the scenario took 483 s. In this time, a transit and a transfer path was planned for each cube and robot arm pair. Fig. 9 shows which planner parametrization was used for individual cubes. For green cubes, movements were planned with two 7 DOF path planners. For yellow cubes movements were planned with the 14 DOF path planner. For red cubes, movements

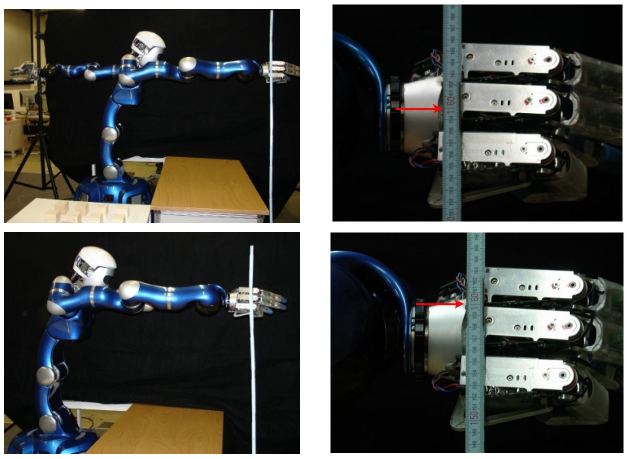


Fig. 8. Sagging of the torso shown with an identical torso configuration and two different configurations of the left arm.

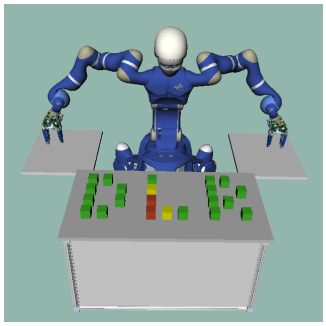


Fig. 9. Shows the goal situation for the task on the real robot. Color encodes which planner parameterization was used to place individual cubes.

could not be planned with a 14 DOF planner because no collision free starting configurations were found, therefore two parallel 7 DOF path planners were used and the paths executed separately. As the color-coding of the cubes shows, the choice of which cube to move next and where to place it, leaves room for optimization through a task planner. Also planning and execution was not yet interleaved. The video accompanying the paper illustrates the execution of the trajectories. A few snapshots are shown in Fig. 10. The video shows that the results of the path optimization for the 14 DOF planner are very unsatisfactory. Path optimization in the 14 DOF configuration space is more difficult since both robot arms are considered simultaneously and can collide with the environment but also with the other arm.

C. Remaining issues

Because of the light weight construction of the robot, the robot arms are not completely independent. The sagging of the torso depends on the configuration of both arms. Thus the deviation cannot be accurately predicted during the planning if the movement of the second arm is not known. While the task elements may be independent the robot components are not. As the start and the goal configuration of the whole robot system is known before the planning starts, the effects of the light weight construction

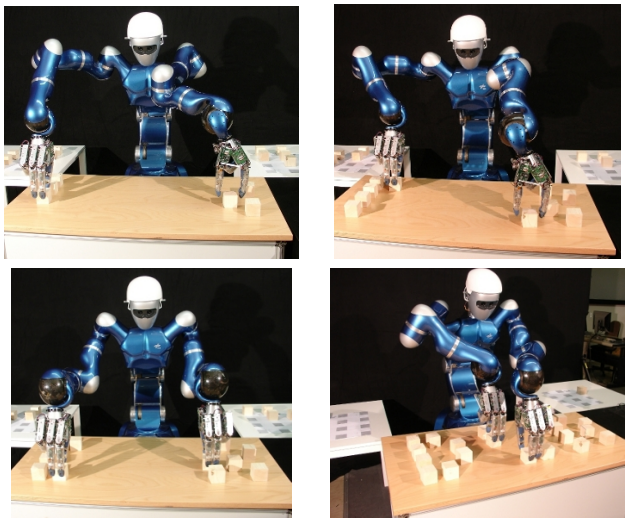


Fig. 10. The real robot placing the cubes of the pattern.

can be accurately modeled at these points. Therefore the disregarded deviations during the movement are small. However, especially if heavier objects are moved with the arms fully extended or in environments with tight narrow passages the remaining position errors may pose problems. As this paper showed, parallelized independent path planning is clearly desirable. Therefore, the issues introduced by the light weight construction definitely need to be addressed in future work. For more rigid robots, the rigid body model is not violated and all recommendation can be directly implemented.

VII. Conclusion

This paper considered aspects of bimanual task execution. In a proof of concept, it was demonstrated that accounting for the functional characteristics during the mapping onto subordinate planners results in speed up of planning and execution of the task. A simple but arbitrarily expandable experimentation scenario was introduced. Tasks are composed of different functional elements. Instead of introducing new elements into a path planner like the weighting of the importance of robot links, the OpenRAVE state of the art path planner implementation was used unmodified and appropriately set up with respect to the functional characteristics of subtasks. It may be argued that the total planning time for the setup is still slow, however it can be greatly improved if the planning step and the execution step on the robot system are interleaved. The next task could already be planned while the current is executed. Also, the collision models could be further simplified and the path optimization process could be improved. At the moment, 50-75% of the planning time are consumed by the path optimizer. The scenario will be extended to more natural scenes including coordinated bimanual tasks. Human-like robot arm configurations and motions are also subject to future work. The task elements itself should be analyzed and ordered by a task planner to achieve an optimized performance. To do this a task planner needs information concerning how difficult a task is or whether a task element is feasible at all. Representations as displayed in Fig. 3 could help to extract this information. The task planner also needs feedback from the subordinate planners. For instance in a cluttered scene where the task goals lie as in Fig. 7 (a) it may well be more efficient to plan and execute the task for each arm separately than to try to find path where both arms can move at once.

VIII. Acknowledgments

The research has been funded by the EC Seventh Framework Programme (FP7) under grant agreement no. 216239 as part of the IP DEXMART. Furthermore the authors would like to thank Rosen Diankov (CMU) for his great support in using OpenRAVE.

References

[1] Y. Guiard, "Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model," *J. Motor Behaviour*, vol. 19, no. 4, pp. 486–517, 1987.

[2] T. Asfour, K. Regenstein, P. Azad, J. Schröder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann, "Armar-III: An integrated humanoid platfrom for sensory-motor control," in *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, 2006, pp. 169–175.

[3] M. Fuchs, C. Borst, P. R. Giordano, A. Baumann, E. Krämer, J. Langwald, R. Gruber, N. Seitz, G. Plank, K. Kunze, R. Burger, F. Schmidt, T. Wimboeck, and G. Hirzinger, "Rollin justin design considerations and realization of a mobile platform for a humanoid upper body," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009.

[4] H. Iwata and S. Sugano, "Design of human symbiotic robot twendy-one," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009.

[5] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and re-grasping tasks," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009, pp. 2464–2470.

[6] R. Diankov, N. Ratliff, D. Ferguson, S. Srinivasa, and J. Kuffner, "Bispace planning: Concurrent multi-space exploration," in *Proc. Int. Conf. on Robotics: Science and Systems*, 2008.

[7] N. Vahrenkamp, A. Barski, T. Asfour, and R. Dillmann, "Planning and execution of grasping motions on a humanoid robot," in *Proc. IEEE Int. Conf. on Humanoid Robots*, 2009.

[8] D. Berenson, J. Chestnutt, S. Srinivasa, J. Kuffner, and S. Kagami, "Pose-constrained whole-body planning using task space region chains," in *Proc. IEEE Int. Conf. on Humanoid Robots*, 2009.

[9] Y. Koga and J. Latombe, "On multi-arm manipulation planning." in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1994, pp. 945–952.

[10] J. Pettré, J.-P. Laumond, and T. Siméon, "A 2-stage locomotion planner for digital actors," in *Proc. Eurographics symposium on Computer Animation (SCA'03)*, 2003.

[11] R. Zöllner, T. Asfour, and R. Dillmann, "Programming by demonstration: Dual-arm manipulation tasks for humanoid robots," in *Proc. IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[12] "OpenRAVE: the open robotics and animation virtual environment," http://openrave.programmingvision.com, 2010, SVN-Revision 1000.

[13] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning." in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2000, pp. 995–1001.

[14] A. Peer, Y. Komoguchi, and M. Buss, "Towards a mobile haptic interface for bimanual manipulations," in *Proc. IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

[15] C. Ott, O. Eiberger, W. Friedl, B. Bäuml, U. Hillenbrand, C. Borst, A. Albu-Schäffer, B. Brunner, H. Hirschmüller, S. Kielhöfer, R. Konietschke, M. Suppa, T. Wimböck, F. Zacharias, and G. Hirzinger, "A humanoid two-arm system for dexterous manipulation," in *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, 2006, pp. 276–283.