Hybrid Motion Planner for a Multi-Armed Robot Performing On-orbit Loco-manipulation Tasks

Ismael Rodríguez*
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Münchner Str. 20, 82234 Weßling, Germany
Technische Universität München
Faculty of Informatics
Garching, Germany
ismael.rodriguezbrena@dlr.de

Sergio Quintero*, Jean-Pascal Lutze,
Peter Lehner, Máximo A. Roa
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Münchner Str. 20, 82234 Weßling, Germany
{Sergio.Quintero, Jean-Pascal.Lutze,
Peter.Lehner, Maximo.Roa}@dlr.de

Abstract—**Large space infrastructure is gaining increasing attention in recent years, including e.g. solar power plants in space or space stations. However, these structures normally are not sent in one piece due to current launch vehicles' size and load capacity limitations. An alternative is to send individual components and subassemblies and perform the final assembly of structures in orbit, mainly using robotic manipulators that execute simple manipulation tasks. However, fixed manipulator systems have limited reachability; therefore, new approaches are required to endow robotic systems with extended mobility and enhanced versatility.**

Several European projects have developed initial prototypes of robotic systems that could execute On-orbit services. In the MOSAR project, a 7 degrees of freedom (DoF) Walking Manipulator (WM) robot performed locomotion and manipulation (loco-manipulation) tasks for the assembly of modular satellites. In this project, the robot and the modules of the satellite were equipped with Standard Interconnects (SI), which provide mechanical, power, and data connectivity for the different components and also enable a straightforward way for their manipulation using a suitable SI as a robotic end effector. A ground scenario was developed to evaluate the system under laboratory conditions. To enhance the reachability and mobility of the WM, the project MIRROR, funded by ESA, pursued the development of a Multi-Armed Robot (MAR) capable of executing loco-manipulation tasks to assemble modular space structures. This system consists of three robotic subsystems, two 7-DoF arms attached to a torso with an extra DoF on its base.

This paper presents a hybrid motion planner for generating trajectories for loco-manipulation tasks using the MAR system. The developed hybrid planner combines a high-level layer that determines the necessary contact states between the robot and the structure, and a low-level layer that plans the joint trajectories among contiguous contact states. The high-level planner implements task-dependent heuristics and a depth-first search approach to reduce the search time for possible contact states. The low-level layer implements an optimization-based motion planner, STOMP, to reduce the random component associated typically with sampling-based motion planners, thus generating as well smoother trajectories. STOMP used as objective function the minimization of joint torques required for executing the different tasks. The performance of the MAR system is evaluated in simulation and on a ground demonstration scenario that simulates the assembly of a modular primary mirror of a space telescope, demonstrating different tasks, including robotic locomotion and manipulation of the mirror parts.

* Equal contribution to this work

Table of Contents

1. Introduction 1
2. Related Work 2
3. Architecture of the Hybrid Planner 2
4. Physical Layer 3
5. Logic Layer 3
6. Experiments 5
7. Conclusions and Future Work 6
References 7
Biography 8

1. Introduction

Recently, on-orbit assembly has become a relevant solution for many space missions requiring large structures. This approach solves the problem of limited size structure, which needs to be fitted in the space rockets [1] since increasing the size of the rockets has a critical impact on the launch price given the extra fuel required. Performing such assemblies is a very difficult task (if not impossible) for astronauts due to the need of performing repetitive actions in a very challenging environment while keeping a suitable working environment for the persons, thus making such missions very expensive and dangerous; therefore, autonomous robotics are required.

Some examples of on-orbit assembly are the MOdular Spacecraft Assembly and Reconfiguration (MOSAR) project and the Prototype for an Ultra Large Structure Assembly Robot (PULSAR) project. In the MOSAR project, a reconfigurable satellite is autonomously built by a Walking Manipulator (WM) with 7 degrees of freedom (DoF), which performs locomotion and manipulation (loco-manipulation) tasks. In our previous work [2] we introduced a prototypical hybrid planner that generates the sequence of tasks that the WM needs to follow to assemble a goal satellite configuration. In the frame of the PULSAR project, where a robotic arm builds a telescope composed of segmented mirror tiles (SMTs), another hybrid planner was developed [3]. In this project, given the weight of the SMTs, it was key for the on-ground demonstrator to minimize the torque perceived in the robotic arm's joints. The MIRROR project pursued the development of a Multi-Armed Robot (MAR) (see Figure 1) with 15 DoFs (two 7 DoFs robotic arms and one extra DoF in the torso) capable of executing loco-manipulation tasks. This project combines the manipulation of SMTs, as in PULSAR, with the required walking strategies in the assembled structure, like in MOSAR.

 DOI: 10.1109/AERO58975.2024.10521183

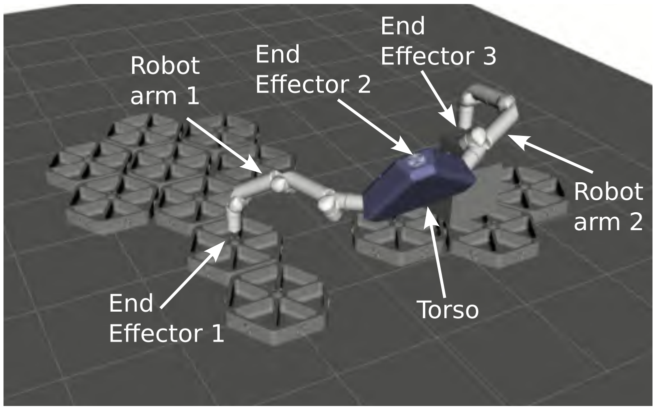


Figure 1: Multi-Armed Robot (MAR). The MAR is composed by two robotic arms and a torso, each of these elements with an end-effector that can interact with any standard interface of the environment.

This work shows the design and evaluation of the hybrid planner, which handles the MIRROR case with the challenges that emerge from interacting with complex elements such as SMTs, plus the loco-manipulation planning for the walking manipulator. A high-level layer determines the robot's task sequence, including the necessary contact states between the robot and the structure. Meanwhile, a low-level layer plans the motions of the joints. In particular, the motions are planned to minimize the joint torques required for executing the different tasks. The performance of the MAR system is evaluated in simulation, performing different tasks such as assembling a modular primary mirror of a space telescope and predefined manipulation tasks like grasping objects with the torso and the robotic arms.

This work is structured as follows. Section 2 includes a summary of related works, mainly focusing on hybrid planners in general and autonomy assembly, specifically in space. Section 3 presents the overall architecture of our solutions, while Sections 4 and 5 present the details of the logic and physical layer, respectively. Experiments and results are introduced in Section 6, and finally, conclusions and future work are described in Section 7.

2. Related Work

Two main topics are covered by the related work: systems using autonomous assembly in space and hybrid planners.

Autonomous assembly in space

Since the early 80s, there have been proposals for On-Orbit Assembly [4], [5], and many missions have followed this path. As mentioned in the introduction, several European projects started the development of On-Orbit Assembly, such as PULSAR [6] and MOSAR [7]. A follow-up project called EROSS IOD is developing phases B2/C for on-orbit servicing missions using the DLR CAESAR robotic arm [8].

Space manipulator systems similar to the CAESAR arm have been constructed for on-orbit servicing missions, such as ETS-VII [9] and Orbital Express [10]. Other manipulators like the Special Purpose Dexterous Manipulator (SPDM) or Dextre [11] have been used for manipulation and servicing tasks. This system installed on the ISS is a dual-arm robot, used for on-orbit servicing in the Robotic Refueling Mission (RRM) [12].

Most of these space-based robotic manipulators can carry out more complex tasks, including autonomous assembly. The MAR system developed for MIRROR is expected to manipulate and assemble different SMTs, relocate Orbit Replaceable Units (ORUs), walk over the structure, and perform maintenance tasks. For such task complexity and variety, a planner is required, and in this work, we present our solution using a hybrid planner.

Hybrid Planners

Hybrid planners are already an established solution for complex robotics since they take the best of two worlds: the fast symbolic planning on the logic layer for task reasoning, and the detailed description of the environment through simulation of the physical layer.

Task planning (carried out in the logic layer) has its basis in transforming geometric constraints into a semantic representation. One of the most known languages for such representations is the Planning Domain Definition Language (PDDL, [13]). However, not all constraints can be easily represented with symbols, so a physical layer is required.

One of the first hybrid planners proposed for simple robotic manipulation tasks was the Asymov system [14]. Many works have considered hybrid planners, also known in the literature as TAMP (Task And Motion Planning), and one of the most relevant exponents is the group of Lozano-Perez and Kaebling at MIT (Massachusetts Institute of Technology). In the last years, they showed that symbols can also be used to show the effect of actions [15]. Later, they learned such effects autonomously to solve more complex tasks [16]. In our previous works with TAMPs, we have learned semantic constraints (applied in the logic layer) by executing the robot tasks in simulation and analyzing the problem that appeared during each specific task [17], [18].

In this work, finding a motion that minimizes the torques perceived in the different joints is critical since the system is tested on ground subject to gravity. In general, two different kinds of approaches can be used for motion planning: either sampling- or optimization-based motion planning. The Probabilistic Roadmap Method (PRM) [19] and the Rapidly-Exploring Random Trees (RRT) [20] are the best examples of sampling-based approaches. Usually, these methods provide a solution fast but sub-optimal, especially if a post-processing step is required to ensure specific path characteristics. On the other hand, optimization-based methods such as TrajOpt [21], covariant gradient-descent (CHOMP) [22], and stochastic gradient-descent (STOMP) [23] can be used to optimize specific characteristics of the path. Given that in-orbit assembly problems are typically not highly constrained, the method employing trajectory optimization is preferred for obtaining robot joint trajectories for the required motions. In particular, STOMP is chosen since with such a method the cost function does not have any constraint on differentiability, which eases the translation of the condition of minimizing the perceived torques on the robot as objective for the planning process.

3. Architecture of the Hybrid Planner

This work uses a solution with the basic structure of a hybrid planner. The input of this planner is the specification of the plan to be executed by the MAR, e.g., assemble a given SMT in a specific position or manipulate a particular object with a specific end effector. As an output of the planner, the commands to the robotic system are generated. These commands include the joint paths to be followed by the MAR and the actions the end effectors have to perform.

As depicted in Figure 2, two main layers compose the planner. The first layer, the logic layer, is in charge of high-level planning. This layer decides on the order and parametrization of tasks to be performed in order to achieve the specified goal. All the reasoning in this layer is purely symbolic, where the positions of the possible locations of the end-effectors do not have any physical meaning, just some reduced representation to know if they are reachable or not. Once a symbolic plan is built, a description of this plan is sent to the next layer, the physical layer. The physical layer is a low-level layer that verifies if the abstract plan designed by the logic layer is executable in the mission's environment. The physical layer informs the logic layer if the task requested is not feasible; in this way, the logic layer can learn from this and generate a new plan. If the requested task is feasible, a motion planner generates a path to execute the task. This path can also be optimized and have some constraints imposed on it to ensure a good result in the actual system. Once these paths are generated, they are given as an output of the planner to be executed in the MAR.

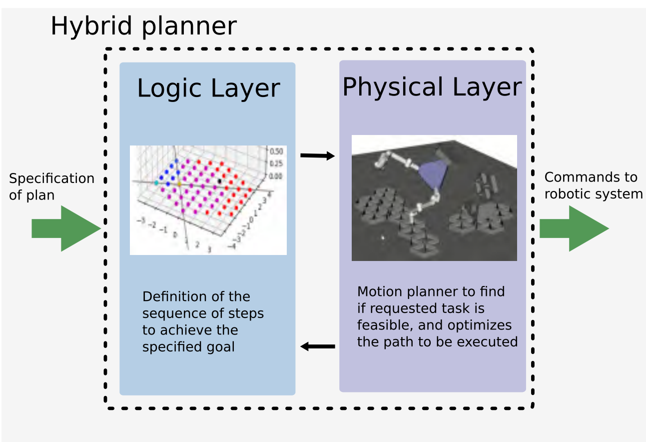


Figure 2: Architecture of hybrid planner. Two layers are the main components, a logic layer and a physical layer. The overall planner receives as input the specification of the plan, and gives as an outcome the commands to be executed in the robotic system.

4. Physical Layer

The physical layer is in charge of the planning of the path of the end effector (EE) of the robot to the goal pose avoiding collisions with other objects in the work-space.

The RRT motion planning approach was implemented to generate the initial trajectory. On the other hand, the STOMP motion planning is used to optimize this initial trajectory. In particular in this work, STOMP uses a torque-dependant cost function to generate smoother trajectories to move the robot in order to reach the goal pose and at the same time to reduce the sum of measured torque in all DoFs.

With either of the implemented approaches the methodology used in the physical layer is the same. For each of the possible next steps that will be evaluated in this layer an initial path for the EE is generated to connect the initial and the goal poses avoiding collisions. After that, an optimization process is applied over the initial path to get a smoother trajectory. This optimized trajectory is evaluated with the cost function shown in Equation 1. This cost function depends on the static torque of each joint calculated at each point of the optimized trajectory. The physical layer returns an optimized path with an associated cost, that in our case is expressed with the following formula:

$$C_\tau(\boldsymbol{\theta_j}) = \sum_{j=0}^{DoF} |\tau_j| \tag{1}$$

Basically, the cost of a given configuration is the sum of the torques measured in each of the DoFs. Note that any other function that receives as an input a given joint configuration would have been also valid. This flexibility on how to choose the cost function is one of the key advantages of STOMP for our application.

Motion planners

When the RRT approach is implemented, the random shortcut optimization algorithm is used. This particular algorithm aims to reduce the path length of a previously generated trajectory.

In contrast, with the STOMP approach the optimization algorithm is used to generate a path that minimizes the desired cost function while preventing collisions. In our implementation, the initial trajectory is a linear interpolation between the initial and the goal poses.

Connection to logic layer

The core of the hybrid planner lies in the connection between the physical layer and the logic layer. As detailed in the following section, the logic layer is in charge of deciding the sequence of the optimal contact points to accomplish the proposed task based on the information provided by the physical layer. In particular, the logical layer calls the physical layer for every possible next contact state to calculate an optimal path and its associated cost between the current configuration and the possible next configuration. With the calculated cost, the logic layer will determine which contact state will be the next in order to reduce the torque consumption in the overall execution of the task. Also, the physical layer discards candidate solutions that may be sent by the logic layer. This information is used to replan accordingly in a higher level by changing the task sequence.

5. Logic Layer

The implemented algorithm in the logic layer finds the necessary contact states between the robot and the structure to create a plan that brings the MAR from a previously defined initial state to a desired final state. It is important to know that the robot has three different end effectors (EE) and the objects that are manipulated are mainly SMTs that have seven connectors known as Standard Interfaces (SI). These SI enable the connection with other mirror tiles and with any of the three EE of the robot.

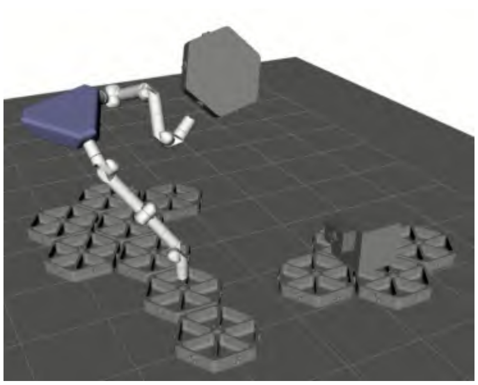
The semantic representation is based on contact states, which are defined by: the EE that is used as base link of the robot, the EE that is attached to the object, the attached object, and the attached SI of the attached object. Therefore, for each state at least two EE's will be attached to an object. Over these contact states the search for a task sequence is done.

The generation of the sequence of transition from the initial to the final state is based on a task-dependent heuristics and a depth-first search approach. In this implementation, the algorithm finds the necessary contact states for the robot to reach the final state.

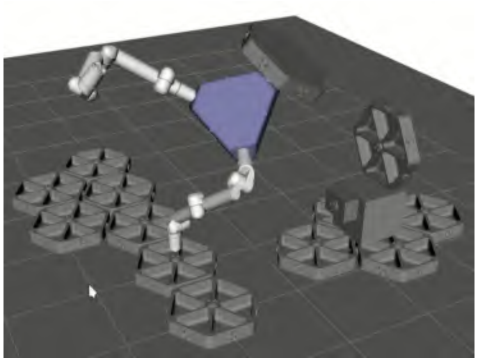
To achieve the desired goal, the contact states are classified as locomotion states and manipulation states. The classification of the states is based on the reachability of the robot as follows. If it is possible for the robot to pick or place the desired object starting in its current position and configuration, then the contact state is classified as a manipulation state. On the other hand, if a manipulation contact state is not reachable, the algorithm calculates a set of possible locomotion states evaluating the tiles of the assembly that can be reached for the robot. In order to be more general in the search of a solution, even if the desired object is reachable, the planner evaluates an additional intermediate locomotion contact state in order to determine if a repositioning of the robot could decrease the torque consumption of the complete sequence of states to accomplish the task.

Manipulation contact states

Different manipulation skills were implemented in the algorithm to increase the versatility of the robot to execute the desired task. In particular, the robot is capable to execute pick and place tasks with its arms and its torso as it is shown in Figures 3a and 3b, respectively.



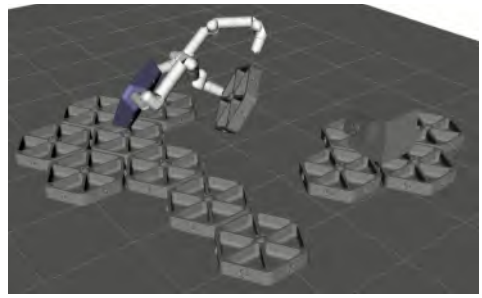
(a) Arm manipulation



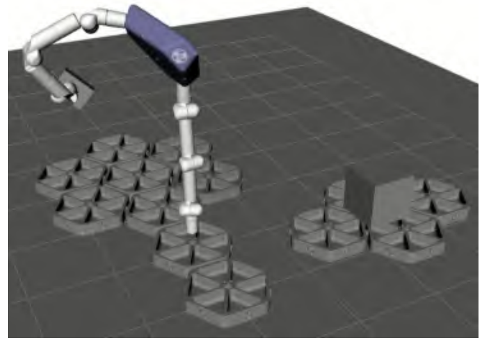
(b) Torso manipulation

Figure 3: Pick and place manipulation skills that the robot can execute using its available limbs.

The algorithm also allows the robot to change the manipulation EE during the execution of a manipulation task, and apart from the SMTs the robot is able to manipulate objects with different geometries if that object has at least one SI, as illustrated in Figures 4a and 4b.



(a) End effector change



(b) Manipulation of a different object

Figure 4: Additional manipulation capabilities of the robot that increase its versatility, allowing it to change the manipulation limb and manipulate objects with different geometries.

Locomotion contact states

The contact states that do not involve the direct manipulation of the desired manipulated body belong to a sequence of locomotion states. This possible locomotion states are determined based on a task-dependent heuristics to reduce the possible next states and accomplish the execution of the proposed task.

Task-dependant heuristics—To determine the best next possible contact state, a task-dependant heuristic is used. The heuristic depends on the goal pose of the object to be manipulated. First, the reachability of the robot is evaluated placing different EE's as base link in each of the SI of the object in its goal pose and verifying this with the physical layer. The obtained set of possible next tiles gives us the possible last locomotion states before placing the object in the desired pose, that is why states closer to the goal are preferred.

After the initial manipulation task of picking the object is reached, a new reachability evaluation is executed. This step evaluates if the robot reaches one of the possible states in the set of possible last locomotion states, otherwise, it is necessary to find additional locomotion contact states. These new locomotion states are created based on a geometric heuristic that assigns a higher relevance to those locomotion states that are in the direction of an imaginary line that connects the current position of the robot with the goal position of the manipulated object.

Algorithm 1 shows this process in the logic layer. In partic-

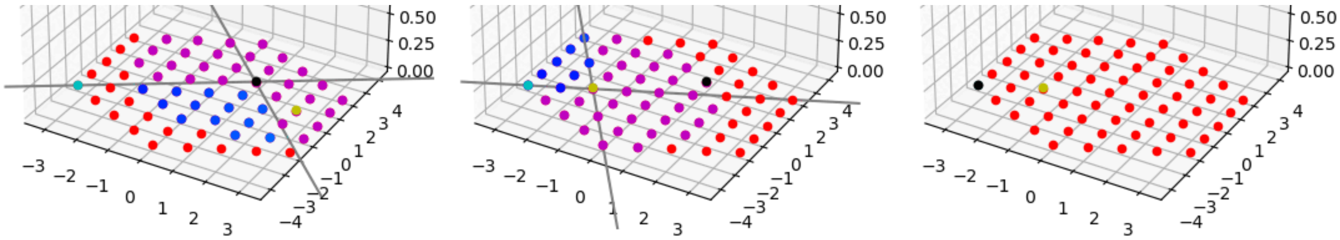


Figure 5: Simplified graphic representation of task-dependent heuristics.

ular, it is possible to appreciate how the algorithm depends mainly on the reachability of the robot to create the initial set of possible next steps, and on the connection with the physical layer to identify the best next contact state. Furthermore, Algorithm 2 determines if the set of next states belongs to the manipulation or locomotion states categories.

Algorithm 1 Logic layer algorithm

$Base \leftarrow Robot.base$
$EE_{manip} \leftarrow Robot.EE_{avail}$
$Pos_{base} \leftarrow Base.pos$
$Config_{init} \leftarrow \mathbf{q}$
$Tile_{base} \leftarrow Tile_0$
$Tile_{ee} \leftarrow Tile_1$
$Obj_{target} \leftarrow SceneObjts[id]$
$Pose_{goal} \leftarrow [\mathbf{X}, \Theta]$
$Obj_{goal} \leftarrow Obj_{target}.setPose(Pose_{goal})$
$Root(Base, EE_{manip}, Tile_{ee})$
$Graph(Root)$
$Task(Obj_{goal})$
for EE in $Robot.EE$ **do**
  for $Conn$ in Obj_{goal} **do**
    $EE.pose \leftarrow Conn.pose$
    $PlaceLLSts.add(getReachStates(EE))$
  for $Conn$ in Obj_{target} **do**
    $EE.pose \leftarrow Conn.pose$
    $PickLLSts.add(getReachStates(EE))$
while $Robot.obj.pose \neq Pose_{goal}$ **do**
  if $Robot.objt = 0$ **then**
    $Goal_{locomotion} \leftarrow PickLLSts$
  else
    $Goal_{locomotion} \leftarrow PlaceLLSts$
  $BstNxtSts \leftarrow locomotionEval(Goal_{locomotion})$
  for BNS in $BstNxtSts$ **do**
    $[path, cost] \leftarrow PhysicalLayer(BNS)$
    $PossNextPath \leftarrow [path, cost]$
  $BestPath \leftarrow GetBestPath(PossNextPath)$
  $MoveRobot(Robot, BestPath)$
  $UpdateRobot(Robot, Obj_{target})$

Figure 5 shows a simplified graphical representation of the implemented heuristics for locomotion contact states in a simple sequence of two steps. In particular, the red dots represent the available SIs in the work space, yellow and black dots represent the available EE of the robot that will be used for locomotion, and the cyan dot represents the desired goal position. First, all reachable SIs by the robot are found, these SIs are represented in the figure as the circle of magenta points. Then, a subset of reachable SIs is chosen as the best subset of next contact steps, this subset is represented in the figure as the zone of blue dots inside the magenta circle of reachable SIs. The limits of this subset are the line that connects the position of the base link of the robot (yellow dot for the first step) and the goal position of the robot (cyan dot), and its perpendicular line. The subset is selected based on the relative distance of the SIs of the subset with respect to the goal position of the robot and the current position of the EE that will be moved to the next contact state.

Algorithm 2 Locomotion evaluation function

function $locomotionEval(Goal_{locomotion})$
  $PossNextStates \leftarrow getReachStates(Base)$
  if Obj_{target} in $PossNextStates.objt$ **then**
    for $NextSt$ in $PossNextStates$ **do**
      $BestNextStates.add(NextSt)$
  if $Goal_{locomotion}$ in $PossNextStates$ **then**
    for $NextSt$ in $PossNextStates$ **do**
      for $LastSt$ in $Goal_{locomotion}$ **do**
        if $NextSt.EE = LastSt.EE$ **then**
          $BestNextStates.add(NextSt)$
  else
    $BestNextStates \leftarrow Heuri(PossNextState)$
  return$BestNextStates$

6. EXPERIMENTS

Three different experiments are performed to validate the proposed planner.

1. The first experiment requests a complex sequence of different tasks that the MAR needs to solve. This sequence of tasks includes the assembly of SMTs, the manipulation of the SMTs between end effectors of the robot, and the manipulation of an ORU. This experiment shows the proper function of the overall system.
2. The second experiment is a comparison between the trajectories given by the RRT and STOMP to effectively compare the reduction of perceived torques in the joints.
3. The third and final experiment is to simulate a failure in one MAR joint. The failure is represented by diminishing the range of movement of the given joint. Such experiment is useful to demonstrate the flexibility of planner by showing how the paths and the task plans are adapted to the situation.

Complex sequence of tasks

The normal behaviour of the planner is covered by this experiment. A request for performing the following tasks is sent to the planner:

- Pick SMT 1 from storage with torso
- Place SMT 1 in designated assembly position
- Pick SMT 2 from storage with arm
- Transfer SMT 2 from arm to arm
- Assemble SMT 2 in designated assembly position

- Pick ORU from storage
- Place ORU in designated position

The requested sequence can be appreciated in Figure 6. In such a sequence the robot is able to perform several manipulation tasks executing its different available skills. In particular, with this sequence the system is capable to manipulate objects with any of its available EE, to transfer objects from one EE to the other when it is allowed by the characteristics of the object (enough SI), and to manipulate objects with different geometries as the prismatic ORU placed in the scene.

Comparison of RRT vs. STOMP

This experiment is intended to verify the proper behaviour of the physical layer. In order to demonstrate this behaviour the above complex set of tasks is performed in two different versions of the physical layer, one using the RRT motion planner and the other one using our approach of STOMP with the previously described cost function.

The MAR is requested to walk five steps to achieve the final goal with its end effector. In both cases the torques during the whole motion are measured and averaged over time, and added up over the 15 joints. The results are presented in Table 1.

Nr. Step	Torque measured along path [Nm]	
	RRT	STOMP
1	3495.8	1511.9
2	2795.8	1836.1
3	2151.6	1904.4
4	283.3	496.4
5	1597.9	1953.6
Total	10324.4	7702.4

Table 1: Simulated torque in each of the paths taken by the MAR to achieve the goal. The value of the table is averaged over time and added up over joints.

The results shown in Table 1 show that a total reduction of more than 25% of the total torques is achieved using the STOMP approach. It may be counter intuitive that for the steps 4 and 5 the RRT approach got better results than STOMP. The cause of this is the initial configuration in which the MAR started both of these steps, that for this particular trajectories had a meaningful impact in the measured torques. Such information can have an impact on the design for the ground demonstrator, since not only the joints may be modified to tolerate the given torques, but the requested tasks can be reconsidered using such information. In particular for this work and this experiment we have opted for minimizing the sum of the torques in each joint, but in the same way other cost functions can be created. In [3] a more complex functions to avoid joint torque limits was generated, meanwhile in [24] the objective was to minimize the disturbance generated in the satellite by the movement of the robot arm.

Limitation of joint range

This experiment shows the capability of the planner to adapt to new unexpected situations. In particular, we simulate the failure of a joint of the MAR, limiting its original movement range from [-0.78, 3.05] radians to [0, 1.88] radians. The affected joint corresponds to the middle joint of the right arm of the robot, as illustrated in Figure 7. Such unexpected situation results in more steps required to execute the proposed complex sequence of task of the first experiment. Figure 8 shows that the STOMP motion planner keeps the affected joint in the new joint range during the execution of the tasks.

Furthermore, the robot requires two more steps to accomplish the tasks due to the new imposed range limitation over the system, as shown in Figure 9. This increment of the required steps occurs because the planner compensates in that way the limitation of the reachable space of the robot. In other words, the robot requires to be closer to the objects and their designated positions in order to execute the manipulation skills. This result shows how the proposed planner is able to adapt the motion plan in the logic layer taking into account the possibles changes that occur in the physical layer.

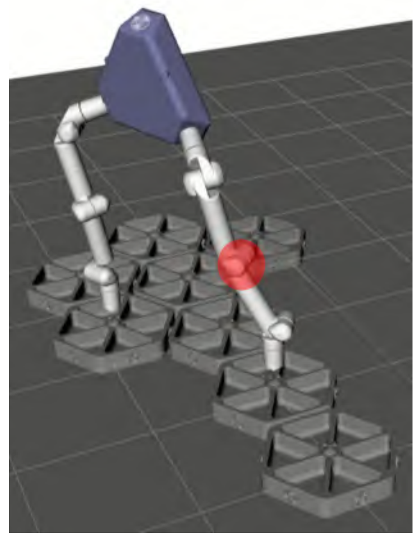


Figure 7: Affected joint of the MAR that changes its movement range from [-0.78, 3.05] to [0, 1.88] radians.

7. Conclusions and Future Work

This work presented a comprehensive hybrid planner for loco-manipulation tasks. A typical complex task solved by the planner involves walking with the MAR, interacting and assembling multiple SMTs, and manipulating an ORU. At the same time, the motions for each task were optimized to minimize the torques perceived in each of the joints, a critical constraint for the ground demonstrator.

The experiments showed the flexibility of the hybrid planner, especially useful for unforeseen scenarios, as it may be in the case of a space mission. As shown in Section 6, a failure was simulated, and the planner could adapt the generated motion and the task sequence. Such capability to adapt to new scenarios is an opportunity to test the critical failures that can invalidate the system to perform specific tasks. All these scenarios can be planned on ground even before the mission takes off to space.

The next steps for the system is to be tested in the ground demonstrator, once it is finished, and corroborate that all simulated results match the actual measurements of the system. This would verify the suitability of the planner to create paths that are executable in the real system, even considering uncertainties coming from the real hardware.

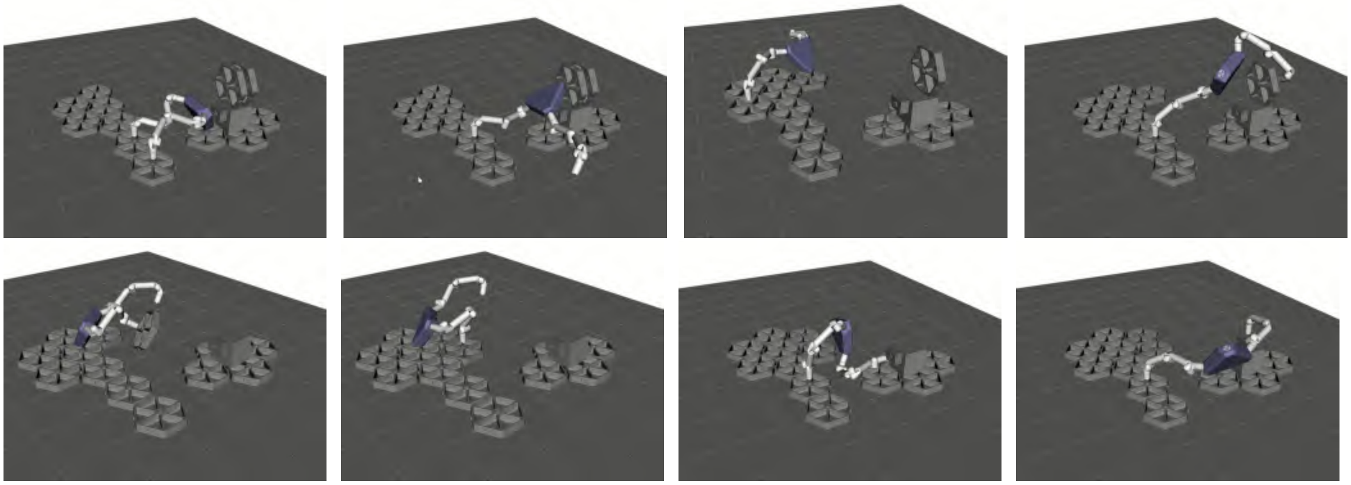


Figure 6: Sequence of the complex task executed by the robot showing the different skills that the system can execute when implementing the proposed motion planner.

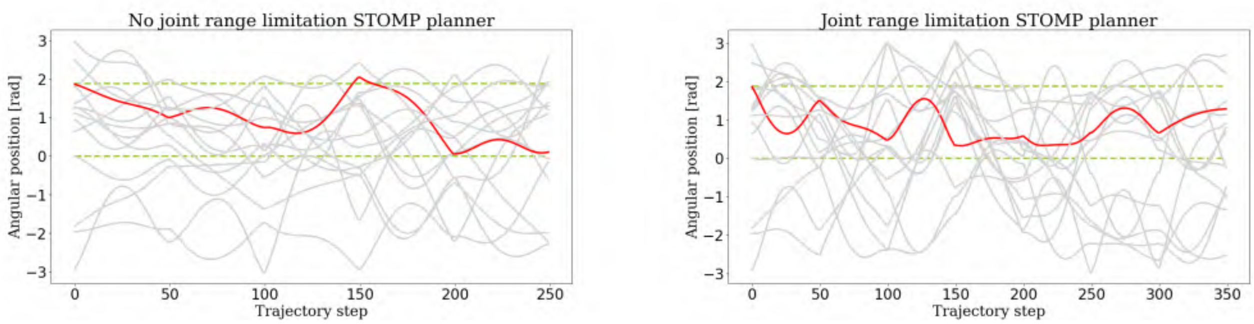


Figure 8: Joint values recorded for the 15 DoF of the MAR. In red is depicted the joint 4 of the right arm. Left: Joint number 4 is NOT limited in range. Right: Joint number 4 is limited in range between 0 and 1.88 radians (show in dashed green). In both cases a STOMP planner is used.

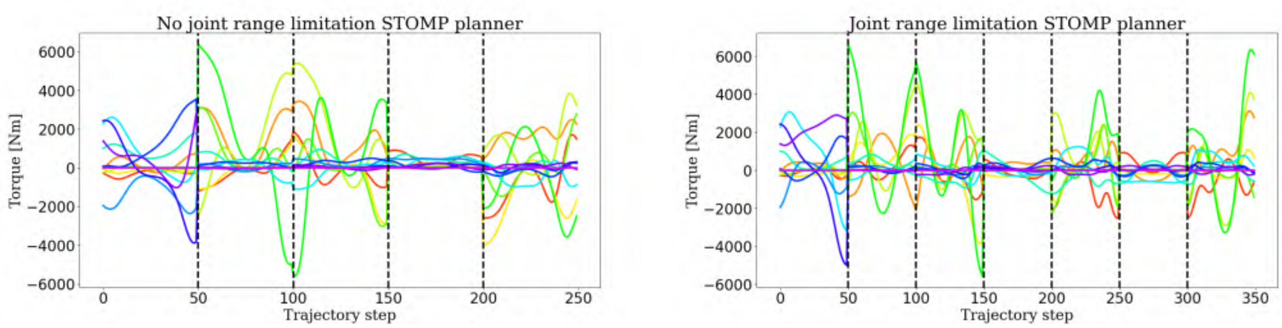


Figure 9: Torque values recorded for the 15 DoF of the MAR. Left: Joint number 4 is NOT limited in range. Right: Joint number 4 is limited in range between 0 and 1.88 radians. In both cases a STOMP planner is used. The black dashed vertical lines indicate the separation between each step taken by the MAR (four in the non limited case, six in the limited case)

REFERENCES

[1] M. Rognant, C. Cumer, J. Biannic, M. A. Roa, A. Verhaeghe, and V. Bissonnette, "Autonomous assembly of large structures in space: a technology review," in *Proc. Europ. Conf. for Aeronautics and Aerospace Sciences (EUCASS)*, 2019.

[2] I. Rodríguez, A. Bauer, K. Nottensteiner, D. Leidner, G. Grunwald, and M. A. Roa, "Autonomous robot planning system for in-space assembly of reconfigurable structures," in *Proc. IEEE Aerospace Conf.*, 2021, pp. 1–17.

[3] J. Martínez-Moritz, I. Rodríguez, K. Nottensteiner, J.-P. Lutze, P. Lehner, and M. A. Roa, "Hybrid planning system for in-space robotic assembly of telescopes using segmented mirror tiles," in *Proc. IEEE Aerospace Conf.*, 2021, pp. 1–16.

[4] D. Akin, M. Minsky, E. Thiel, and C. Kurtzman, "Space applications of automation, robotics and machine in-

telligence systems (ARAMIS) - phase II," in *NASA contractor report 3734*, 1983.

[5] K. Yoshida, "Space Robot Dynamics and Control: To Orbit, From Orbit, and Future," in *Robotics Research*, J. Hollerbach and D. Koditschek, Eds. Springer, 2000, pp. 449–456.

[6] V. Bissonnette, C. Bazerque, S. Trinh, C. Porte, G.Arcin, M. Rognant, J.-M. Biannic, C. Cumer, T. Loquen, and X. Pucel, "A simulation tool for in-orbit assembly of large structures," in *Proc. Int. Symp. on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2020.

[7] P. Letier, X. Yan, M. Deremetz, A. Bianco, G. Grunwald, M. Roa, R. Krenn, M. Munoz, P. Dissaux, J. Garcia, R. Ruiz, L. Filippis, G. Porceluzzi, M. Post, M. Walshe, and P. Perryman, "MOSAR: MOdular Spacecraft Assembly and Reconfiguration demonstrator," in *Proc. Symp. Advanced Space Technologies Robotics and Automation (ASTRA)*, 2019.

[8] A. Beyer, G. Grunwald, M. Heumos, M. Schedl, R. Bayer, W. Bertleff, B. Brunner, R. Burger, J. Butterfaß, R. Gruber, *et al.*, "CAESAR: Space robotics technology for assembly, maintenance, and repair," in *Proc. Int. Astronautical Congress (IAC)*, 2018.

[9] K. Yoshida, K. Hashizume, and S. Abiko, "Zero reaction maneuver: flight validation with ETS-VII space robot and extension to kinematically redundant arm," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2001, pp. 441–446.

[10] A. Ogilvie, J. Allport, M. Hannah, and J. Lymer, "Autonomous satellite servicing using the orbital express demonstration manipulator system," in *Proc. Int. Symp. on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2008.

[11] E. Coleshill, L. Oshinowo, R. Rembala, B. Bina, D. Rey, and S. Sindelar, "Dextre: Improving maintenance operations on the International Space Station," *Acta Astronautica*, vol. 64, pp. 869–874, 2009.

[12] L. Metcalfe and T. Hillebrandt, "Robotic refuelling mission - demonstrating satellite refuelling technology on board the ISS," in *Proc. Int. Symp. on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2014.

[13] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "PDDL-the planning domain definition language," *AIPS98 Planning Competition Committee*, 1998.

[14] F. Gravot, S. Cambon, and R. Alami, "aSyMov: a planner that deals with intricate symbolic and geometric problems," in *Int. Symp. on Robotics Research*. Springer, 2005, pp. 100–110.

[15] T. Lozano-Pérez and L. P. Kaelbling, "A constraint-based method for solving sequential manipulation planning problems," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2014, pp. 3684–3691.

[16] L. P. Kaelbling and T. Lozano-Pérez, "Learning composable models of parameterized skills," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2017, pp. 886–893.

[17] I. Rodríguez, K. Nottensteiner, D. Leidner, M. Durner, F. Stulp, and A. Albu-Schäffer, "Pattern recognition for knowledge transfer in robotic assembly sequence planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3666–3673, 2020.

[18] I. Rodríguez, K. Nottensteiner, D. Leidner, M. Kaßecker, F. Stulp, and A. Albu-Schäffer, "Iteratively refined feasibility checks in robotic assembly sequence planning," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1416–1423, 2019.

[19] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[20] S. M. LaValle, "Rapidly exploring dense trees," *Planning Algorithms*, pp. 228–237, 2004.

[21] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization," in *Robotics: Science and systems*, 2013, pp. 1–10.

[22] M. Zucker, N. Ratliff, A. Dragan, M. Pivtoraiko, M. Klingensmith, C. Dellin, A. Bagnell, and S. Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning," *Int. J. Robotics Research*, vol. 32, no. 9–10, pp. 1164–1193, 2013.

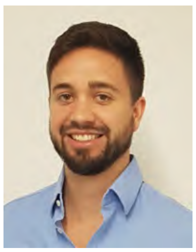
[23] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2011, pp. 4569–4574.

[24] I. Rodríguez, J.-P. Lutze, H. Mishra, P. Lehner, and M. A. Roa, "Hybrid planning to minimize platform disturbances during in-orbit assembly tasks," in *2022 IEEE Aerospace Conference (AERO)*. IEEE, 2022, pp. 1–11.

BIOGRAPHY

Ismael Rodríguez *received his degree of Ingeniero en Electrónica from Universidad ORT Uruguay in 2015. In 2023 he received his Ph.D. in Robotics at the Technical University Munich for his research done at the German Aerospace Center (DLR) in Weßling. His research is focused on the development of an assembly planner that is aligned with the requirements of the mass customization phenomenon.*

Sergio E. Quintero Benavides *received his degree of Ingeniero Mecatrónico from Universidad Nacional de Colombia in 2023. In 2022 he joined to the Institute of Robotics and Mechatronics at the German Aerospace Center (DLR) where he focused his work in motion planing of multi-armed robots. Since 2023 he is pursuing his Master degree in Advance robotics at the Warsaw University of Technology and Keio University in Tokyo.*



Jean-Pascal Lutze *received his B.Eng degree in Robotic and Automation from Heilbronn University of Applied Sciences. Since 2017, he is part of the Institute of Robotics and Mechatronics at the German Aerospace Center (DLR). Since 2019 he is pursuing his Master degree in Robotics, Cognition, Intelligence at the Technical University of Munich (TUM). His main focus is space robotics.*

Peter Lehner *is a researcher at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), since 2014. He received his master's degree in Computer Engineering from the Technical University of Berlin in 2014. His current research activities include developing methods for autonomous motion generation for mobile manipulation systems to empower mobile robots to autonomously interact with their environment.*

Máximo A. Roa-Garzón *received his doctoral degree in 2009 from Universitat Politecnica de Catalunya, and the Project Management Professional (PMP) Certification in 2016. He worked for Hewlett Packard R&D before joining the Institute of Robotics and Mechatronics of the German Aerospace Center (DLR) in 2010 as Senior Research Scientist, leading the group on Robotic Planning and Manipulation.*