



**US Army Corps
of Engineers®**
Engineer Research and
Development Center



DamBot™

Integrating MOVEit Motion Constraints on a Novel Robotic Manipulator

Benjamin Christie, Jordan Klein, Anton Netchaev,
and Garry Glaspell

November 2023

The US Army Engineer Research and Development Center (ERDC) solves the nation's toughest engineering and environmental challenges. ERDC develops innovative solutions in civil and military engineering, geospatial sciences, water resources, and environmental sciences for the Army, the Department of Defense, civilian agencies, and our nation's public good. Find out more at www.erdclibrary.on.worldcat.org/discovery.

To search for other technical reports published by ERDC, visit the ERDC online library at www.erdclibrary.on.worldcat.org/discovery.

Integrating MOVEit Motion Constraints on a Novel Robotic Manipulator

Jordan Klein and Anton Netchaev

*US Army Engineer Research and Development Center (ERDC)
Information Technology Laboratory (ITL)
3909 Halls Ferry Road
Vicksburg, MS 39180-6199*

Benjamin Christie and Garry Glaspell

*US Army Engineer Research and Development Center (ERDC)
Geospatial Research Laboratory (GRL)
7701 Telegraph Road
Alexandria, VA 22315-3864*

Final Technical Report (TR)

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

Prepared for Headquarters, US Army Corps of Engineers
Washington, DC 20314-1000

Under PE 29785D, Project Number 497149, Task Number 8, “Dambot™”

Abstract

MOVEit, a widely used Robot Operating System framework, plans composite tasks, where the high-level sequence of actions is fixed and known in advance. However, these tasks need to be tailored and adapted to the environmental context. This framework uses custom trajectory planners, known as controllers, to solve goals that are fully defined within the configuration space. Libraries, such as the Open Motion Planning Library, provide a collection of motion planners that can solve task-space goals. An exact spatial and joint replication of the robotic manipulator's mechanics, typically Universal Robot Description Format and Semantic Robot Description Format files, is required. Common arms such as the Panda-Manipulator and OpenMANIPULATOR-X provide these files in their respective public repositories, but custom arms require significant modification or even a complete rewrite of these files.

DISCLAIMER: The contents of this report are not to be used for advertising, publication, or promotional purposes. Citation of trade names does not constitute an official endorsement or approval of the use of such commercial products. All product names and trademarks cited are the property of their respective owners. The findings of this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

DESTROY THIS REPORT WHEN NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

Contents

Abstract	ii
Figures and Tables.....	v
Preface.....	vi
1 Introduction.....	1
1.1 Background.....	1
1.2 Objectives.....	2
1.3 Approach and Scope	2
2 Research and Discussion	6
2.1 Instructions	6
2.1.1 Installation.....	6
2.1.2 Launch.....	6
2.2 Utilizing Movement Constraints	7
2.3 Controlling a Physical Robotic Manipulator	9
2.4 Custom hdt_nav Package	10
2.5 Known Issues.....	12
2.6 Future Work	13
3 Conclusion.....	15
References	16
Appendix : Configuration and Launch Files.....	18
A.1 Package Configuration Files.....	18
A.1.1 CMakeLists.txt.....	18
A.1.2 package.xml	19
A.2 Source Files, Header Files, and Scripts.....	20
A.2.1 hdt_nav_node.cpp	20
A.2.2 hdt_nav_node.h	20
A.2.3 hdt_nav_controller.h.....	21
A.2.4 hdt_nav_controller.cpp.....	22
A.2.5 hdt_nav/quaternion.h.....	24
A.2.6 pose-nav-array.py	25
A.2.7 scan_pattern.py	27
A.3 URDF Properties and State Files.....	29
A.3.1 /hdt_arm.srdf.....	29
A.3.2 hdt_arm_positionctrl.urdf	34
A.3.3 hdt_arm_effortctrl.urdf	52
A.3.4 hdt_arm_properties.xacro	70
A.4 Launch Files	74

A.4.1	hdt_arm_bringup_1.launch.....	74
A.4.2	spawn_model.launch.....	77
A.4.3	hdt_nav_node.launch.....	77
Abbreviations.....		78
Report Documentation Page (SF 298).....		79

Figures

Figures

1. Fifteen-foot HDT arm spawned into Rviz.....	3
2. Simulated arm collapsed around origin.....	4
3. The HDT arm spawning successfully in Gazebo.	5
4. HDT rest position left and unfolding right.....	10
5. Camera attached to arm in simulation.	13

Listing

1. Sample control using MOVEitCommander on the command line.	9
---	---

Preface

This study was conducted for the US Army Corps of Engineers under Program Element 29785D, Project Number 497149, Task Number 8, “Dambot™.” The technical monitor was Dr. Anton Netchaev.

The work was performed by personnel from the US Army Engineer Research and Development Center, Information Technology Laboratory (ERDC-ITL) and the Data Representation Branch of the Topography Imagery and Geospatial Research Division, ERDC Geospatial Research Laboratory (GRL). At the time of publication, Mr. Vineet Gupta was branch chief; Mr. Michael Mailloux was acting division chief; and Dr. Austin Davis was the technical director for the Geospatial Research Laboratory. The deputy director of ERDC-GRL was Ms. Valerie L. Carney, and the director was Mr. David R. Hibner. The deputy director of ERDC-ITL was Dr. Jackie S. Pettway, and the director was Dr. David A. Horner.

The authors would like to acknowledge the following individuals for their contributions to this project: Mr. Chuck Ellison and Mr. Steven Bunkley.

The commander of ERDC was COL Christian Patterson, and the director was Dr. David W. Pittman.

1 Introduction

1.1 Background

MOVEit (Coleman et al. 2014) is a widely used Robot Operating System (ROS) framework that describes and plans composite tasks using a sequence of actions at a high level and individual joint and kinematic control at a low level (Gorner et al. 2019). The environmental context of these plans is defined in the realm of dynamic planning scenes. This means that it is possible to plan motions in either the Cartesian space or the joint space. MOVEit uses custom trajectory planners, known as controllers, to solve goals. These goals are usually fully defined within the configuration space. Libraries, such as the Open Motion Planning Library (OMPL), provide a collection of motion planners (Motion Planners available in *MOVEit!—Planners Benchmarking* documentation) that can solve tasks in the space, assuming that a description of the robotic manipulator is provided (Soetens et al. 2020). Specifically, the description contains both Unified Robot Description Format (URDF) (ROS 2020) and Semantic Robot Description Format (SRDF) (ROS wiki 2019) files. Common arms such as Panda-Manipulator (ROBOTIS 2023) and OpenMANIPULATOR-X (GmbH 2020) provide these files in their respective public repositories, but custom arms require these files to be rewritten to their specific geometry.

MOVEit provides a roscpp interface and a Python application programming interface (API) known as `moveit_commander`. This report discusses both, but production code should be written using the roscpp interface. While being faster and having more functionality, the roscpp interface is able to control the robot more finely than the Python API can. MOVEit has recently integrated constrained motion control in the Cartesian space using the OMPL (Maeyer 2020; Kingston et al. 2018). This greatly improves the mapping potential of robotic manipulators using sensors such as depth cameras or lidars. Historically, the best way to maintain a position or orientation *constraint* was to create a waypoint plan with a very high number of points. However, this methodology did not actually constrain movement. For movements near singularities, the planner would typically navigate away from the point, not holding any of the desired *constraints*. Incorporating OMPL has resolved several of these issues.

1.2 Objectives

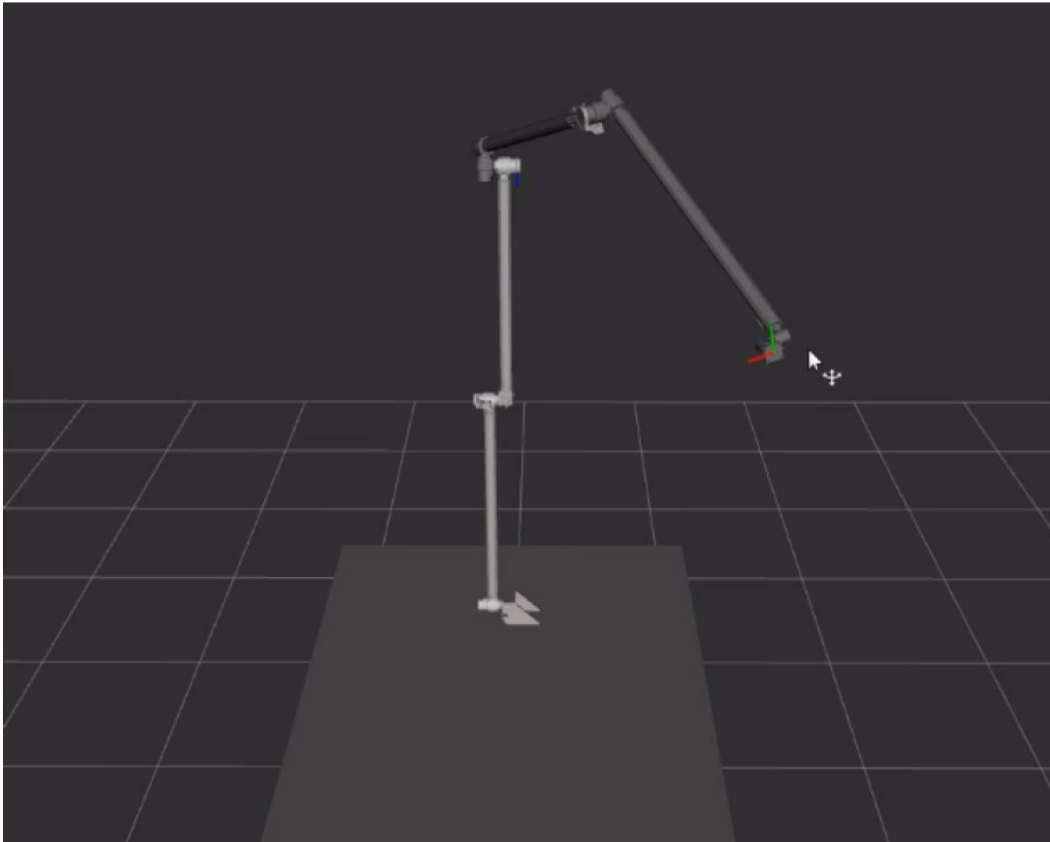
The US Army Corps of Engineers (USACE) owns, maintains, and operates over 700 large-scale dams with a growing number exceeding their design life. Inspecting the outlet works of the earth dams is hazardous for human inspectors. The USACE Engineer Research and Development Center (ERDC) is developing a solution that is capable of performing detailed autonomous inspections while an operator teleoperates a robot. To inspect the outlet works we plan on using a fifteen-foot robot arm equipped with a camera. The use of a robotic arm will reduce the risk to personnel and allow a rapid assessment. This research supports the “Revolutionize and Accelerate Decision Making” research and development priority, which states,

USACE leverages data science, artificial intelligence, machine learning, robotics, trade space analytics, geospatial engineering, and advanced models and simulations to revolutionize geospatially informed decision making by understanding the operational environment, predicting likely outcomes, decreasing uncertainties by 80%, and shaping faster decisions that minimize risk, reduce costs, and accelerate mission delivery by 5× for military, civil works, and disaster response operations.

1.3 Approach and Scope

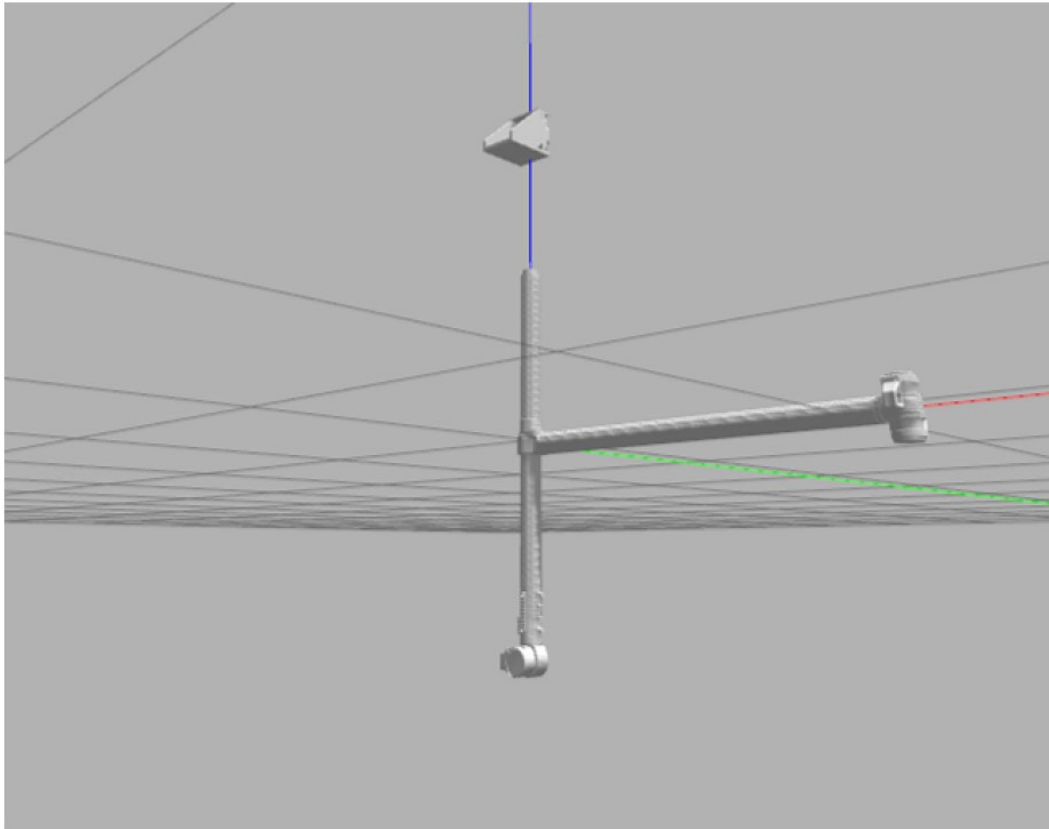
In this report, we leverage a repository provided by the manufacturer of the robotic arm, HDT Global. The xacro file, provided by HDT Global, was used by MOVEit for motion planning, inverse kinematics (IK), and forward kinematics. If the xacro file is formatted properly, then it can also be used for simulation. MOVEit compiles the xacro file into a URDF file at runtime, which is typically loaded to the robot description parameter. This process works well inside Rviz (a 3D visualization tool for ROS) (ROS wiki 2018) or in the real world. Figure 1 demonstrates the the robot description parameter loaded into the ROS param server and then spawned into Rviz. However, more configuration is required if the user wants to operate the arm in a physics simulation such as Unity (Unity-Technologies 2021) or Gazebo (ROS wiki 2021).

Figure 1. Fifteen-foot HDT arm spawned into Rviz.



There are a few issues with the base HDT repository when integrating with Gazebo. Specifically, the robot semantic description parameter that MOVEit uses to deduce relative joint distances resulted in all the links, in the Gazebo model, collapsing around the origin, as shown in Figure 2. Although the cause of this bug is not fully known, Gazebo likely thinks that the joint distances are *absolute*, not *relative*. Note that when the connection between Gazebo and Rviz is working correctly, the joint states of the arm should be published from Gazebo, not Rviz. Rviz should simply be mirroring what Gazebo is showing.

Figure 2. Simulated arm collapsed around origin.

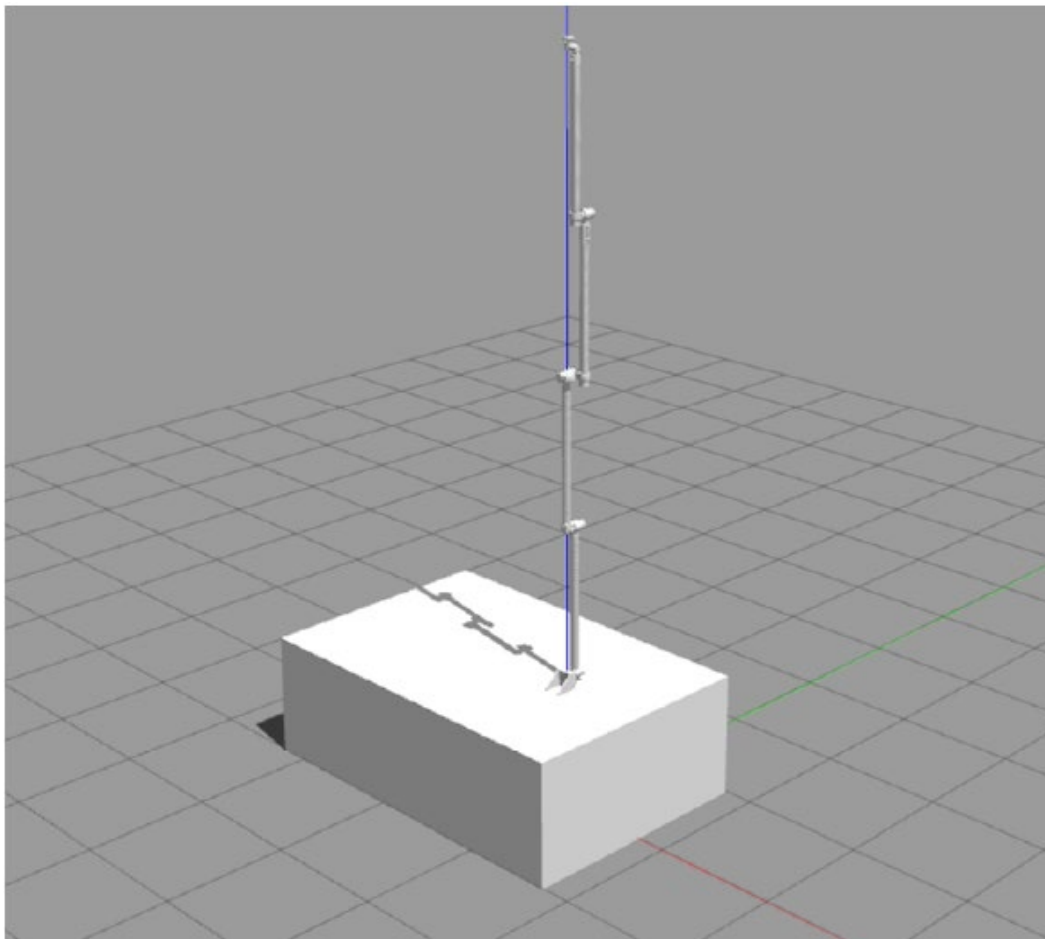


Additionally, the Gazebo model did not have any transmission tags. As a result, the Gazebo joints would not move. These tags were added manually. Both a velocity controller version and a position controller version are shown in the Appendix. The type of controller detailed in the transmission tags does not affect MOVEit. However, it will affect how the simulated Gazebo model behaves. Strangely, if a velocity controller is used, the links still sometimes collapse to the origin when the robot semantic description parameter is loaded to the ROS param server. This does not always occur but should be avoided. Since MOVEit requires the robot semantic description to function, the position-based controller should be used.

Another issue within the repository is the incomplete description of moments of inertia of the arm. Specifically, to map in simulation, the moments of inertia of the robot arm must be as accurate as possible in order to predict and control movements. These exact moments of inertia of every link in the arm were not provided, so an accurate reconstruction of the arm in Gazebo was impossible. Fortunately, one can mitigate the erroneous vibration of the arm in simulation by setting the moment of

inertia of the arm to approximate values, which was done for the final model. Note that Gazebo does not take into account these moments of inertia for the real arm; they seem to affect only the physics of the arm in simulation. An HDT arm working in Gazebo is shown in Figure 3.

Figure 3. The HDT arm spawning successfully in Gazebo.



Virtually no changes to the MOVEit planner or the IK configuration of the HDT arm were needed. The default settings provided by HDT seem to work well. It should be noted that in the future, TRAC-IK should be used for IK to achieve better solution speeds. TRAC-IK is an improvement of the Orocos Kinematics and Dynamics Library (KDL) IK solver that MOVEit uses by default (Beeson and Ames 2015). Unfortunately, we were unable to get TRAC-IK to perform IK for the arm group using the `moveit_commander` API or the `roscpp` interface. TRAC-IK may need to be built directly into the MOVEit configuration for the HDT arm. This is best done using the `moveit_setup_assistant` package.

2 Research and Discussion

2.1 Instructions

This section details how to launch and use the HDT arm and the custom hdt_nav package. Note that the hdt_nav package is intended to be an introduction to MOVEit and will change in the future. The constraint-based planning in MOVEit is still in its infancy; further testing is recommended before field use.

2.1.1 Installation

To install and use the packages outlined in this report, follow the instructions below:

1. Install ROS Melodic.
2. Install all required libraries and packages:

```
Sudo apt install ros-melodic-moveit* ros-melodic gazebo-ros-pkgs  
ros-melodic-gazebo-ros-control
```
3. Copy the files in the Appendix to the /catkin_ws/src folder. The package configuration files and source files, header files, and cripts are required for compilation. The URDF properties and state files are required to run in simulation. The launch files control the arm.
4. Build the workspace:

```
cd ~/catkin_ws && catkin_make
```

2.1.2 Launch

To launch the files included in this report, perform the following:

1. Launch the HDT arm according to their ReadMe:

```
roslaunch hdt_nav hdt_arm_bringup_1.launch simulation:=true
```

This will launch the HDT arm using the new URDF and xacro files listed in the Appendix.
2. If launching in simulation, the gazebo model is spawned using the spawn_model.launch file listed in the Appendix.
3. Assuming that the base of the arm group is parallel to the floor plane, launch the hdt_nav node provided in the Appendix or by calling

```
roslaunch hdt_nav hdt_nav_node
```

The `hdt_nav` node is not necessary for manual control. However, when moving the physical arm, it is always recommended to have terminal access. If the arm begins moving unexpectedly, terminate the node. When the `roscpp` node is terminated, the HDT arm brakes and holds its position. A physical emergency stop switch is also highly recommended for use in the field.

2.2 Utilizing Movement Constraints

Before the implementation of movement constraints in MOVEit, the motion plans were generated by the Covariant Hamiltonian Optimization for Motion Planning (CHOMP) (Ratliff et al. 2009) and Stochastic Trajectory Optimization for Motion Planning (STOMP) (Williams, n.d.) planners. However, CHOMP and STOMP had difficulty accurately controlling the orientation and position of the arm for the duration of the motion (motion planners available in *MOVEit!—Planners Benchmarking* documentation; Maeyer 2020). Specifically, if we wanted the end effector to hold a constant orientation as it moved, that is, parallel to the ground plane, the built-in waypoint planner resulted in a choppy motion that did not hold points to a relative pose. With the incorporation of movement constraints into OMPL, it is now possible to maintain a desired orientation or position constraints along the robotic arm. This is achieved by passing a lambda function to the OMPL motion planners. This lambda function is included in the optimization of the path produced. The movement constraints are not perfect since (most) arms do not have infinite degrees of freedom (DoF). Constraining a point of the robotic arm essentially reduces the DoF available for movement. Thus, a range of error must be passed with the constraint parameters so that the generated motion plans are smoother and the robotic arm can avoid singularities.

This report only discusses using the movement constraints from the `roscpp` interface. In order to use the MOVEit movement constraints in `roscpp` code, perform the following:

1. Verify that the ROS node is spinning (running).
2. Verify that the `moveit::planning_interface::PlanningSceneInterface` is initialized.
3. Verify that the `moveit::planning_interface::MoveGroupInterface` is initialized.

4. Use the `moveit_msgs::OrientationConstraint` message to construct a complete orientation constraint message. Make sure to set the weight of the message to 1.0.
5. Push the message to a `moveit_msgs::Constraints` vector using the orientation constraints handle.
6. Finally, set `pathConstraints` by calling the `setPathConstraints` method of the global `MoveGroupInterface` variable.

There are a few potential pitfalls with this approach. First, we need to be sure that the interfaces referenced are unique to the group; otherwise, MOVEit will hang. This is achievable with the following code block using boost libraries to send a reference of the `MoveGroupInterface` to the callback function:

```
void callback(const MsgType::ConstPtr& msg,
             MoveGroupInterface& mgi) { ... }
int main(void) {
    // setup ROS node and initialize mgi, psi
    nh.sub(topicname,
          sz_t, boost::bind(callback, _1, boost::ref(mgi)));
    // finish initializing ROS node and start spinning
}
```

As a result, the `MoveGroupInterface` global variable can be referenced in the callback function without relying on a global variable, which is a concern for larger projects. Note that the normal roscpp callback function only accepts one variable. The same process can be performed to include the `PlanningSceneInterface` if that is desired. The attached code presents a solution that passes a reference to a structure that contains both interfaces and ways to control the arm group. An example of moving to a pose goal is shown below. Note that the path's execution is separate from its planning. Thus, you can plan a route, analyze its cost, and determine whether you should execute it.

```
geometry_msgs::Pose goalPose = null;
// set goalPose to desired pose of end effector
moveit::planning_interface::MoveGroupInterface
moveGroup{PLANNING_GROUP};
Plan my_plan;
moveGroup.setPoseTarget(goalPose);
moveGroup.plan(my_plan);
bool success = (moveGroup.move() == moveit::planning_interface::MoveItErrorCode::SUCCESS);
```

Moving to a joint goal is technically less complicated since MOVEit always plans in the joint space. However, the API is slightly more complex, as shown below:

```
std::vector<double> goalJointStates = something;
// set goalJointState to desired state of PLAN-
NING_GROUP group
moveit::planning_interface::MoveGroupInterface moveGroup{PLAN-
NING_GROUP};
moveit::planning_interface::MoveGroupInterface::Plan my_plan;
moveit::core::RobotStatePtr current_state = moveGroup.getCur-
rentState();
const robot_state::JointModelGroup*
joint_model_group =
    moveGroup.getCurrentState()->getJointModelGroup (PLAN-
NING_GROUP);
std::vector<double> joint_group_positions; cur-
rent_state->copyJointGroupPositions(
joint_model_group, joint_group_positions); moveGroup.set-
JointValueTarget(goalJointStates); moveGroup.plan(my_plan);
bool success = (moveGroup.move() == moveit::
planning_interface::MoveItErrorCode::SUCCESS);
```

2.3 Controlling a Physical Robotic Manipulator

Controlling a real-world robotic manipulator using MOVEit typically requires additional configuration. Fortunately, HDT Global's repository for their custom arm runs out of the box. If a different robotic arm is used, it is necessary to verify that the ROS drivers that operate the joints integrate into the MOVEit hardware interface. If the ROS drivers do not interact with `ros_control`, then writing a custom implementation that provides low-level hardware control is necessary. The details of such implementations are outside the scope of this report. However, it must be done in accordance to the `ros_control` API. This will allow you to control the robot arm using a `JointGroupPositionController` from the position controllers group. The next step is to integrate the arm into MOVEit using `moveit_setup` assistant, provided that accurate URDF files are available.

Listing 1. Sample control using MOVEitCommander on the command line.

```
import rospy, moveit_commander, geometry_msgs.msg
mg = moveit_commander.MoveGroupCommander("arm")
rospy.init_node("cmdLineNode")
pose = geometry_msgs.msg.Pose
### fill in data in the pose message
mg.set_pose_target(pose)
plan = mg.go(wait=True) # wait=True is important
mg.stop()
mg.clear_pose_targets()
```

Note that it is preferable to create a node instead of using the command line, but the `moveit_commander` package is still useful when simple control is needed. The `hdt_nav` package uses `roscpp` to create a node to control the arm. While most testing of this package was performed in simulation, the `hdt_nav` package also worked on the real HDT arm. Figure 4 shows the arm unfolding from its rest position using the `hdt_nav` package. All the launch files are provided in the Appendix.

Figure 4. HDT rest position (*left*) and unfolding (*right*).



2.4 Custom `hdt_nav` Package

This package essentially acts as a wrapper for moving the HDT arm. It is not a final solution to incorporate simultaneous localization and mapping (SLAM) problems into MOVEit, but it is a good place to start for future projects that involve the HDT arm and orientation-constrained motion. The implementation is relatively simple, but the features it uses are novel. The control flow for this package is as follows.

This package provides many useful motions, including rotating the end effector about the axis, moving along planes, and scanning walls. This package assumes that the obstacle space is properly constructed. It has no knowledge that is external to MOVEit's interfaces, so the SLAM module must communicate with the planning scene interface. Implementing new

motions can be done using the aforementioned interfaces. Additionally, a simple quaternion calculation library is provided. This library is simpler than tf2's quaternion library and is much lighter weight. It is not mandatory that this library be used, but it may be useful in the future.

There are many improvements that can be made to this package, including integrating signed distance fields (SDFs). The `ros_industrial` package has integrated the Yak library for truncated SDFs. Simply put, a truncated SDF is a probabilistic representation of a solid surface in a 3D space (Canelhas 2017). The Yak library implements a motion planner that can use truncated SDFs to map an object using a “next best view” approach, as implemented in the `nbvplanner`. Thus, provided an SDF in real time, hardcoded motions would not need to be programmed ahead of time. Instead, MOVEit would dynamically adapt to the environment. Existing packages would need to be fully rewritten for such an algorithm to be feasible in a real-time environment. This can be done using SDFs or a voxel-based approach. Note that due to the limited and nonspherical structure of robotic arms' workspaces (when taking singularities into account), special care needs to be taken when marking what voxels should be ignored. Some voxels may appear to be mappable but may result in the arm being in a singularity. The SDF approach may mitigate this issue. Additionally, the use of a precomputed IK package such as IKFast may fix this since the robot workspace can be fully known beforehand (IKFast: The Robot Kinematics Compiler 2011). All the scripts used to control the arm are provided in the Appendix.

Algorithm 1: Flow of hdt Navigation Package

```

Ensure:  $X$  is stowed
1:  $X \leftarrow \text{ready}$ 
2:  $X_i \leftarrow \text{ready}$ 
3:  $dz \leftarrow 0.05$ 
4:  $dy \leftarrow 0.25$ 
5:  $\text{offset} \leftarrow dz, dy$ 
6:  $C_{\text{orientation}} \leftarrow Q_{\text{parallel}}$ 
7: start mapping
8: while mapping do
9:  $X \leftarrow X_i + \text{offset}$ 
10:  $X \leftarrow X_i - 2 * \text{offset}$ 

```

11: $offset \leftarrow offset + dz$

12: end while

2.5 Known Issues

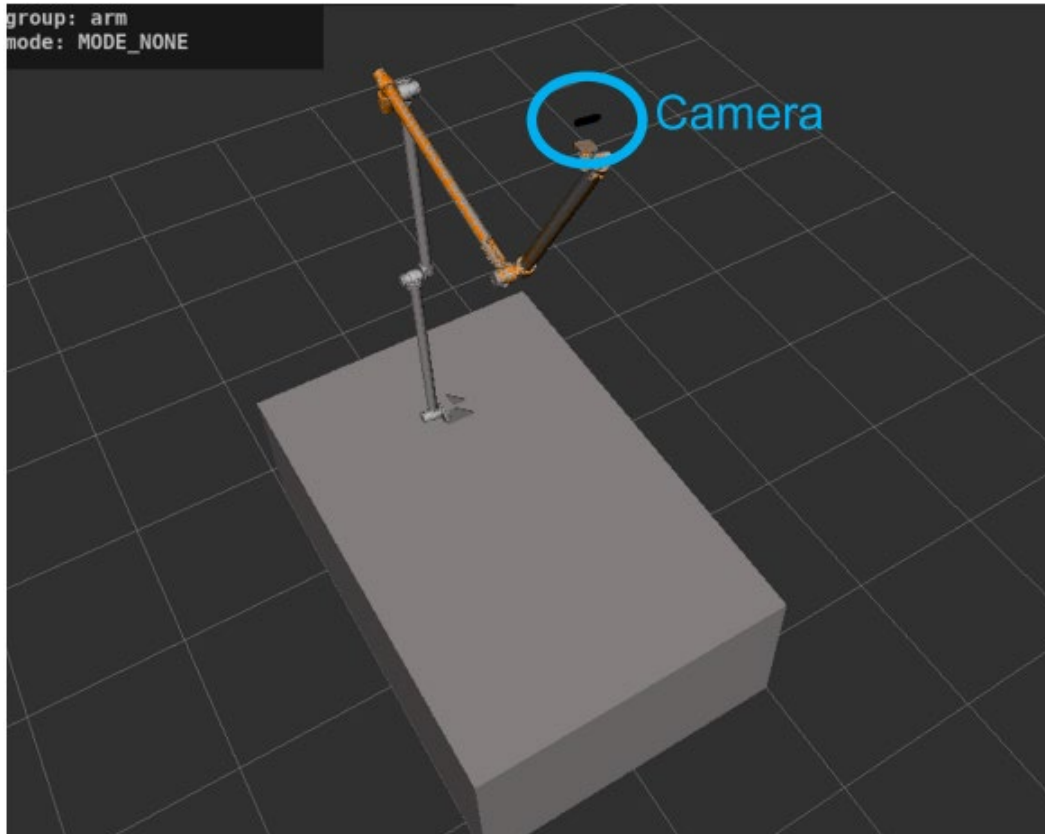
There are a few issues with the HDT repository and the `hdt_nav` package that future programmers should be aware of:

- The configuration that HDT provided of their move group does not interact well with subscribers. Any attempt to subscribe from a `roscpp` file results in a synchronization issue. This is true for both simulation and nonsimulation time. A solution for this has not yet been found.
- The HDT arm has joint restrictions due to the gigabit Power over Ethernet cable internal to the entire length of the arm. This can effect motions to and away from the stow position in order to protect the cable. *When controlling the HDT arm through Rviz, always verify that the motion plan presented by MOVEit does not collide with unmapped obstacles.* If the map generated by SLAM is accurate, this should not be an issue.
- MOVEit always uses *doubles* instead of *floats*. Never use *floats* when programming with MOVEit.
- Motion constraints are not cleared after a motion plan has been planned and executed. You must manually clear them after every motion plan.
- The Eigen3 library has some linking issues that affect MOVEit. If building with catkin does not work, when including the Eigen3 library, reinstalling MOVEit may help.
- None of the launch files provided by HDT work, aside from the bringup files. This includes all gazebo files and most move group files. When using gazebo, follow the instructions listed in section Section 2.1.2 and use the launch files from the `hdt_nav` package.
- Assuming that the robot starts in the stow position, it is unlikely that the robot will discover where the ceiling is using the end effector camera. Thus, it may be prudent to set an obstacle matching the ceiling height in the URDF file. This object may also be added to the MOVEit planning scene by determining the height of the room prior to launch time.

2.6 Future Work

In this report, we use the information that MOVEit provides to determine the position of the scanning sensor. Since each joint position is known, the pose of each link, from the kinematic chain, can be determined with high precision. Therefore, the pose of the end effector, relative to some fixed position near the base of the robot arm, is known within some margin of error. Vibration, sag, and gear slack are the main concerns. This information can be used by SLAM, via a transform tree, to construct a real-time map of the environment using the robot arm. We have demonstrated this in simulation with a Panda-Manipulator and physically on a custom OpenMANIPULATOR-X arm, leveraging Real-Time Appearance-Based Mapping (Labbé and Michaud 2019) to perform SLAM. We were successful mapping the environment with the smaller OpenMANIPULATORX robot arm; we are curious to see how the aforementioned vibration, sag, and gear slack affect the longer HDT arm. To date, we have explored adding a camera to the virtual HDT arm, shown in Figure 5, and plan to do the same with the physical arm.

Figure 5. Camera attached to arm in simulation.



Additionally, the map created by SLAM can be fed back into MOVEit's motion planning scene for collision avoidance. Obstacles could be added to MOVEit's planning scene using the motion planning scene interface, which is initialized at launch time. This is configured using the `moveit_setup_assistant` package. Furthermore, since the map generated by SLAM is constantly changing, the planning scene will dynamically update, and the motion plans generated by MOVEit will change accordingly.

3 Conclusion

It is clear that advances in the OMPL and MOVEit aid in mapping. Path constraints, although limited to orientation, position, and visibility, can be used to generate trajectories that can quickly map civil work structures. Furthermore, the HDT robotic arm repository can be used out of the box in real life. With modifications, the repository can also be used in simulation.

In the future, providing a framework for arbitrary constraints (represented by a lambda function) will be the next step for SLAM using robotic arms. Although the constraints provided in MOVEit are promising, they are also very limited. Using solely OMPL and movement constraints make mapping nonplanar structures difficult. More work will be necessary to provide variable and complex movement constraints.

References

- Beeson, P., and B. Ames. 2015. "TRAC-IK: An Open-Source Library for Improved solving of Generic Inverse Kinematics." In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)* (Seoul, South Korea) Nov: 928–35. <https://doi.org/10.1109/HUMANOIDS.2015.7363472>.
- Canelhas, D. R. 2017. "Truncated Signed Distance Fields Applied to Robots." *Örebro Studies in Technology* 76, p. 161.
- Coleman, D., I. A. Sucan, S. Chitta, and N. Correll. 2014. "Reducing the Barrier to Entry of Complex Robotic Software: A MOVEit Case Study." *Journal of Software Engineering for Robotics* 1 (5): 3–16. <https://doi.org/10.6092/JOSER.2014.05.01.p3>.
- Franka Emika GmbH. 2020. "PANDA'S Instruction Handbook." Accessed 27 July 2021. https://dentec.pl/pliki/Artykul/855_dentec-frankainstrukcja-uzytownika.pdf.
- Gorner, M., R. Haschke, H. Ritter, and J. Zhang. 2019. "MOVEit! Task Constructor for Task-Level Motion Planning." *2019 International Conference on Robotics and Automation (ICRA)* (May): 190–96. <https://doi.org/10.1109/ICRA.2019.8793898>.
- IKFast: The Robot Kinematics Compiler. 2011. Accessed 9 August 2022. http://openrave.org/docs/latest_stable/openravepy/ikfast/#ikfast-the-robot-kinematics-compiler.
- Kingston, Z., M. Moll, and L. E. Kavraki. 2018. "Sampling-Based Methods for Motion Planning with Constraints." *Annual Review of Control, Robotics, and Autonomous Systems* 1 (1): 159–85. <https://doi.org/10.1146/annurev-control-060117-105226>.
- Labbé, M., and F. Michaud. 2019. "RTAB-Map as an Open-Source Lidar and Visual Simultaneous Localization and Mapping Library for Large-Scale and Long-Term Online Operation." *Journal of Field Robotics* 36 (2): 416–46. <https://doi.org/10.1002/rob.21831>.
- Maeyer, J. 2020. "A New Approach for Planning with Path Constraints in MOVEit." <https://moveit.ros.org/moveit/2020/09/10/ompl-constrained-planninggsoc.html>.
- Ratliff, N., M. Zucker, J. A. Bagnell, and S. Srinivasa. 2009. "CHOMP: Gradient Optimization Techniques for Efficient Motion Planning." *2009 IEEE International Conference on Robotics and Automation* (Kobe) (May): 489–94. <https://doi.org/10.1109/ROBOT.2009.5152817>.
- ROBOTIS. 2023. "OpenMANIPULATOR-X." Accessed 27 July 2021. <https://emanual.robotis.com/docs/en/platform/openmanipulatorx/overview/>.
- ROS. 2020. "urdf." <https://github.com/ros/urdf>.

ROS wiki. 2018. “Rviz.” <http://wiki.ros.org/rviz>.

ROS wiki. 2019. “srdf.” <https://wiki.ros.org/srdf>.

ROS wiki 2021. “Gazebo.” [http://wiki.ros.org/gazebo ros pkgs](http://wiki.ros.org/gazebo_ros_pkgs).

Shadow Robot Company. 2018. “Motion Planners Available in MOVEit!—Planners Benchmarking Documentation.” Accessed 28 July 2021. [https://planners-benchmarking.readthedocs.io/en/latest/user guide/2 motion planners.html](https://planners-benchmarking.readthedocs.io/en/latest/user_guide/2_motion_planners.html).

Soetens, P., T. Issaris, H. Bruyninckx, S. Joyeux, and R. Smits. 2020. “Orocos Project Documentation.” *Orocos*. <https://docs.oroocos.org>.

Unity-Technologies. 2021. “Unity-Robotics-Hub.” <https://github.com/Unity-Technologies/Unity-Robotics-Hub>.

Williams, J. n.d. “STOMP: Stochastic Trajectory Optimization for Motion Planning—Autonomous Motion.” *Max Planck Institute for Intelligent Systems*. Accessed 28 July 2021. <https://is.mpg.de>.

Appendix A: Configuration and Launch Files

This appendix lists the files in the hdt_nav package and the files modified from the original HDT repository.

A.1 Package Configuration Files

A.1.1 CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0.2)
project(hdt_nav)
set(CMAKE_EXPORT_COMPILE_COMMANDS ON)

## Compile as C++11, supported in ROS Kinetic and newer
#add_compile_options(-std=c++11)

## Find catkin macros and libraries
## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)
## is used, also find other catkin packages
find_package(catkin REQUIRED COMPONENTS
  geometry_msgs
  moveit_msgs
  std_msgs
  moveit_ros_planning_interface
  rviz_visual_tools
  moveit_visual_tools
  roscpp
  rospy
)

#####
## catkin specific configuration ##
#####
## The catkin_package macro generates cmake config files for your
package
## Declare things to be passed to dependent projects
## INCLUDE_DIRS: uncomment this if your package contains header
files
## LIBRARIES: libraries you create in this project that dependent
projects also need
## CATKIN_DEPENDS: catkin_packages dependent projects also need
## DEPENDS: system dependencies of this project that dependent
projects also need
catkin_package(
  INCLUDE_DIRS include
)

## Specify additional locations of header files
## Your package locations should be listed before other locations
include_directories(
  include
  ${catkin_INCLUDE_DIRS}
)
```

```

## Declare a C++ executable
## With catkin_make all packages are built within a single CMake
context
## The recommended prefix ensures that target names across
packages don't collide
add_executable(${PROJECT_NAME}_node src/hdt_nav_node.cpp
include/hdt_nav/hdt_nav_controller.cpp)

## Specify libraries to link a library or executable target
against
target_link_libraries(${PROJECT_NAME}_node
    ${catkin_LIBRARIES}
)

## Mark executables for installation
## See
http://docs.ros.org/melodic/api/catkin/html/howto/format1/buildin
g_executables.html
install(TARGETS ${PROJECT_NAME}_node
    RUNTIME DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
)

```

A.1.2 package.xml

```

<?xml version="1.0"?>
<package format="2">
  <name>hdt_nav</name>
  <version>0.1.4</version>
  <description>The hdt_nav package</description>

  <maintainer email="benc00@vt.edu">Benjamin
Christie</maintainer>

  <license>TODO</license>

  <buildtool_depend>catkin</buildtool_depend>
  <build_depend>geometry_msgs</build_depend>
  <build_depend>moveit_msgs</build_depend>
  <build_depend>roscpp</build_depend>
  <build_depend>rospy</build_depend>
  <build_export_depend>geometry_msgs</build_export_depend>
  <build_export_depend>moveit_msgs</build_export_depend>
  <build_export_depend>roscpp</build_export_depend>
  <build_export_depend>rospy</build_export_depend>
  <exec_depend>geometry_msgs</exec_depend>
  <exec_depend>moveit_msgs</exec_depend>
  <exec_depend>roscpp</exec_depend>
  <exec_depend>rospy</exec_depend>

</package>

```

A.2 Source Files, Header Files, and Scripts

A.2.1 hdt_nav_node.cpp

```
#include <hdt_nav/hdt_nav_controller.h>
#include <hdt_nav/hdt_nav_node.h>

void pointCallback(const geometry_msgs::Point::ConstPtr &msg,
                  HDTNavController &navCont) {
    geometry_msgs::Pose p = navCont.getCurrentPose();
    ROS_INFO("Current pose : %f %f %f", p.position.x,
p.position.y, p.position.z);
    p.position.x += msg->x;
    p.position.y += msg->y;
    p.position.z += msg->z;
    // goPointPlanar(movegroup, *msg);
    navCont.goPoseGoal(p);
}

int main(int argc, char **argv) {
    ros::init(argc, argv, "hdt_nav_node");
    ros::NodeHandle node_handle;
    ros::AsyncSpinner spinner(1);
    spinner.start();
    HDTNavController cont{};
    cont.goJointGoal(bentHome);
    geometry_msgs::Point cp = cont.getCurrentPose().position;
    geometry_msgs::Point pa;
    geometry_msgs::Point pb;
    pa.x = cp.x;
    pa.y = cp.y - 1.0;
    pa.z = cp.z - 1.0;
    pb.x = cp.x;
    pb.y = cp.y + 1.0;
    pb.z = cp.z + 1.0;
    cont.rotateEE(1, 0, 0, 0.5 * PI_F);
    cont.rotateEE(1, 0, 0, -0.5 * PI_F);
    cont.goJointGoal(bentHome);
    cont.moveAlongWall(pa, pb);
    ros::Subscriber sub =
node_handle.subscribe<geometry_msgs::Point>(
    "goal_point", 1000,
    boost::bind(
        pointCallback, _1,
        boost::ref(cont))); // pass the movegroup as an
arg to the callback
    ros::waitForShutdown(); // since already using an async
spinner
    return 0;
}
```

A.2.2 hdt_nav_node.h

```
#ifndef HDT_NAV_NODE_H
#define HDT_NAV_NODE_H
#define PI_F 3.14159265358979323846f
#define SQRT2_2 0.70710678118f
```

```

#include <moveit/move_group_interface/move_group_interface.h>
#include
<moveit/planning_scene_interface/planning_scene_interface.h>

#include <moveit_msgs/DisplayRobotState.h>
#include <moveit_msgs/DisplayTrajectory.h>

#include <moveit_msgs/AttachedCollisionObject.h>
#include <moveit_msgs/CollisionObject.h>

#include <hdt_nav/quaternion.h>
#include <tf2_geometry_msgs/tf2_geometry_msgs.h> // an excellent
quaternion implementation
#include <moveit_visual_tools/moveit_visual_tools.h>

const std::vector<double> zero{7, 0.0}; // initialize a vector of
len 7 with 0.0
const std::vector<double> home{0.5 * PI_F, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0};
const std::vector<double> bentHome{0.3 * PI_F, 0.0, 0.0, -0.4 *
PI_F,
    0.5 * PI_F, -0.2 * PI_F, 0.0};
const tf2::Quaternion parallelQuaternion(0.0, SQRT2_2, SQRT2_2,
0.0);

#endif

```

A.2.3 hdt_nav_controller.h

```

#ifndef _HDTNAVCONTROLLER
#define _HDTNAVCONTROLLER
#include <hdt_nav/hdt_nav_node.h>

class HDTNavController {
private:
    const std::string PLANNING_GROUP = "arm";
    const std::string ENDPOINT_LINK = "endpoint_link";
    const std::string BASE_LINK = "base_link";
    moveit::planning_interface::MoveGroupInterface
moveGroup{PLANNING_GROUP};
    moveit::planning_interface::PlanningSceneInterface
planningSceneInterface;
public:
    HDTNavController() = default;
    geometry_msgs::Pose getCurrentPose();
    bool goPointPlanar(const geometry_msgs::Point&
goalPoint);
    bool goJointGoal(const std::vector<double>&
goalJointStates);
    bool goPoseGoal(const geometry_msgs::Pose& goalPose);
    bool moveAlongWall(const geometry_msgs::Point& pointA,
const geometry_msgs::Point& pointB);
    bool rotateEE(int X, int Y, int Z, double rot); // TODO
may overload this with (quaternion) if necessary

};
#endif

```

A.2.4 hdt_nav_controller.cpp

```

#include <hdt_nav/hdt_nav_node.h>
#include <hdt_nav/hdt_nav_controller.h>
#include <hdt_nav/quaternion.h>

geometry_msgs::Pose HDTNavController::getCurrentPose() {
    return moveGroup.getCurrentPose().pose;
}

bool HDTNavController::rotateEE(int X, int Y, int Z, double rot)
{
    geometry_msgs::Pose cp = getCurrentPose();
    Quaternion q(cp.orientation);
    q.rotate(X, Y, Z, rot);
    cp.orientation = q.toGMQ();
    // done with setup, goal pose == cp
    return(goPoseGoal(cp));
}

bool HDTNavController::moveAlongWall(const geometry_msgs::Point&
pointA, const geometry_msgs::Point& pointB) {

    geometry_msgs::Pose cp = getCurrentPose();
    cp.position = pointA;
    goPoseGoal(cp);
    cp = getCurrentPose();
    // go from cp to pointA. then, scan from pointA(x, y) ->
pointB(x, y) along pointA(z). Then, increment z by dz
    int resolution = 10; // ten sweeps, this can be changed
    double dz = (pointB.z - pointA.z) / resolution;
    double dx = (pointB.x - pointA.x);
    double dy = (pointB.y - pointA.y);
    geometry_msgs::Point np;
    // assuming that cp == pointA
    for (; resolution > 0; resolution--) { // while(resolution--
> 0)
        np = getCurrentPose().position;
        np.x += dx;
        np.y += dy;
        goPointPlanar(np); // go to pointB(x, y)
        np.z += dz;
        np.x -= dx;
        np.y -= dy;
        goPointPlanar(np); // go to pointA(x, y, z+dz)
    }
    return true;
}

bool HDTNavController::goPointPlanar(const geometry_msgs::Point&
goalPoint) {

    moveit::planning_interface::MoveGroupInterface::Plan gPlan;
    // first, get current EE orientation.
    geometry_msgs::Quaternion quat =
moveGroup.getCurrentPose().pose.orientation; //

```

getEndEffectorLink() is assumed, can use ENDPOINT_LINK if this breaks

```

    ROS_INFO("Current goal quaternion %f %f %f %f", quat.x,
    quat.y, quat.z, quat.w);

    // note that this is not being used to its fullest extent.
    constraints are one of moveit's best features for high dof arms
    moveit_msgs::OrientationConstraint ocm;
    ocm.link_name = ENDPOINT_LINK;
    ocm.header.frame_id = BASE_LINK;
    ocm.orientation = quat;
    ocm.absolute_x_axis_tolerance = 0.35;
    ocm.absolute_y_axis_tolerance = 0.35;
    ocm.absolute_z_axis_tolerance = 0.35;
    ocm.weight = 1.0;
    moveit_msgs::Constraints constraints;
    constraints.orientation_constraints.push_back(ocm);
    moveGroup.setPathConstraints(constraints);
    // done setting orientation constraint
    geometry_msgs::Pose gPose;
    gPose.position = goalPoint;
    gPose.orientation = quat;

    moveGroup.setPoseTarget(gPose);
    moveGroup.setPlanningTime(20.0); // planning with constraints
    can be painfully slow (double for loop)
    moveGroup.plan(gPlan);
    bool success = (moveGroup.move() ==
    moveit::planning_interface::MoveItErrorCode::SUCCESS);
    // be sure to undo any constraints!
    moveGroup.clearPathConstraints();
    return success;
}

bool HDTNavController::goJointGoal(const std::vector<double>&
goalJointStates) {

    moveit::planning_interface::MoveGroupInterface::Plan my_plan;
    // RobotState is the object that contains all the current
    position/velocity/acceleration data.
    moveit::core::RobotStatePtr current_state =
    moveGroup.getCurrentState();
    const robot_state::JointModelGroup* joint_model_group =
    moveGroup.getCurrentState() -
    >getJointModelGroup(PLANNING_GROUP);
    // Next get the current set of joint values for the group.
    std::vector<double> joint_group_positions;
    current_state->copyJointGroupPositions(joint_model_group,
    joint_group_positions);

    moveGroup.setJointValueTarget(goalJointStates);
    moveGroup.plan(my_plan);
    bool success = (moveGroup.move() ==
    moveit::planning_interface::MoveItErrorCode::SUCCESS);

```

```

        return success;
    }

bool HDTNavController::goPoseGoal(const geometry_msgs::Pose&
goalPose) {
    // navigation to a pose with no tolerance restrictions
    moveit::planning_interface::MoveGroupInterface::Plan my_plan;
    moveGroup.setPoseTarget(goalPose);
    moveGroup.plan(my_plan);
    bool success = (moveGroup.move() ==
moveit::planning_interface::MoveItErrorCode::SUCCESS);
    ROS_INFO("result : %i", success);
    return success;
}

```

A.2.5 hdt_nav/quaternion.h

```

#ifndef QUATERNION_H_
#define QUATERNION_H_
#include <geometry_msgs/Quaternion.h> // useful for some
definitions
#include <tf2_geometry_msgs/tf2_geometry_msgs.h> // useful for
some definitions

#include <cmath>

class Quaternion {
private:
    double x = 0.0;
    double y = 0.0;
    double z = 0.0;
    double w = 1.0;

    void normalize() {
        double mag_factor = 1.0 / std::sqrt(x * x + y * y + z
* z + w * w);
        x = x * mag_factor;
        y = y * mag_factor;
        z = z * mag_factor;
        w = w * mag_factor;
        return;
    }

public:
    Quaternion(){};
    Quaternion(const geometry_msgs::Quaternion& q) : x(q.x),
y(q.y), z(q.z), w(q.w) {
        normalize(); // should be entirely unnecessary
    }
    Quaternion(geometry_msgs::Quaternion&& q){
        Quaternion(std::move(q)); // should be unnecessary,
since a constructor for const &q is defined
    };
    Quaternion(const tf2::Quaternion& q) : x(q.getX()),
y(q.getY()), z(q.getZ()), w(q.getW()) {
        normalize(); // should be entirely unnecessary
    }
}

```

```

        Quaternion(tf2::Quaternion&& q){
            Quaternion(std::move(q)); // should be unnecessary,
since a constructor for const &q is defined
        };
        Quaternion(double X, double Y, double Z, double W) :
x(X), y(Y), z(Z), w(W) {
            normalize(); // in case it is initialized stupidly
        }
        geometry_msgs::Quaternion toGMQ() {
            geometry_msgs::Quaternion q;
            q.x = x;
            q.y = y;
            q.z = z;
            q.w = w;
            return q;
        }

        Quaternion &operator*(const Quaternion &a) {
            x = (y * a.x) + (x * a.y) + (z * a.w) - (w * a.z);
            y = (x * a.z) - (y * a.w) + (z * a.x) + (w * a.y);
            z = (x * a.w) + (y * a.z) - (z * a.y) + (w * a.x);
            w = (x * a.x) - (y * a.y) - (z * a.z) - (w * a.w);
            normalize();

            return *this;
        }
        Quaternion &operator*(const geometry_msgs::Quaternion
&gmQ) {
            (*this) * Quaternion(gmQ);
            return *this;
        }

        Quaternion &rotate(int X, int Y, int Z,
            double rot) {
            double factor = std::sin(0.5 * rot);
            Quaternion helperQ(X*factor, Y*factor, Z*factor,
std::cos(0.5*rot));
            helperQ.normalize();

            (*this) * helperQ;

            return *this;
        }
    };

#endif // QUATERNION_H_

```

A.2.6 pose-nav-array.py

```

#!/usr/bin/python

import rospy
import geometry_msgs.msg
import moveit_msgs.msg
import moveit_commander
import copy

```

```

import sys

# This function initializes the node. It is called by
# running 'python <this-node>.py' or by calling
# roslaunch
def main():
    global move_group
    move_group = moveit_commander.MoveGroupCommander("arm") #
    this is defined globally for ease of use
    rospy.init_node("pose_nav_array_sub", anonymous= True)
    sub = rospy.Subscriber("/pose_array", geometry_msgs
.msg.PoseArray, poseArrayCallback);
    print("Current position is : ")
    print(move_group.get_current_pose().pose) rospy.spin() #
    essentially a while loop, but other
    functions can be called asynchronously, such as the callback
    function

# Callback function for the pose array goal topic. #
@param message : the message received on the
    pose_array topic
# if the message.data is empty, then nothing happens def
poseArrayCallback(message):
    for pose in message.poses: print("Currently planning
        for pose : ") print(pose)
        move_group.set_pose_target(pose)
        plan = move_group.go(wait=True)
        move_group.stop() # this ensures that there is
        no residual movement move_group.clear_pose_targets() # i
        think this
        SHOULD be unnecessary, but I was having issues with it
        going back to the home position after every goal if I
        didn't do this

    return # end

def manual_control():
    z_dir_diff = float(input("Enter z dir dif : "))
    x_dir_diff = float(input("Enter x dir dif : "))
    y_dir_diff = float(input("Enter y dir dif : ")) n_points
    = int(input("Enter number of points : "))
    zincr = z_dir_diff / n_points yincr
    = y_dir_diff / n_points xincr =
    x_dir_diff / n_points
    waypoints = []
    wpose = move_group.get_current_pose().pose for n in
    range(1, n_points+1):
        cpose = wpose cpose.position.x += n
        * xincr cpose.position.y += n *
        yincr cpose.position.z += n * zincr
        waypoints.append(copy.deepcopy(cpose)
        )
    (plan, fraction) = move_group.

```

```

        compute_cartesian_path(waypoints, 0.01, 0.0)
        move_group.execute(plan, wait=True)

if name == " main ":
    main()

```

A.2.7 scan_pattern.py

```

#!/usr/bin/python
import rospy
from geometry_msgs.msg import Quaternion
import moveit_commander
from math import pi as PI
from math import sin, cos
from numpy import array
# This function initializes the node. It is called by
# running 'python <this-node>.py' or by calling roslaunch
def main():
    global move_group
    move_group = moveit_commander.MoveGroupCommander("
arm") # this is defined globally for ease of use
    rospy.init_node("pose_nav_array_sub", anonymous= True)
    move_group.go(bent_home, wait=True)
    cpose = move_group.get_current_pose().pose
    cpose.position.y += 1.0 move_group.go(cpose, wait=True)
    cpose.orientation = quat_rotate(cpose.orientation,
-PI / 2, 'y')
    cpose.position.y += 0.15
    cpose.position.x += 0.25
    move_group.go(cpose, wait=True) # goto rotated point
    move_group.go(bent_home, wait=True)
    cpose = move_group.get_current_pose().pose
    cpose.position.y -= 1.0 move_group.go(cpose, wait=True)
    cpose.orientation = quat_rotate(cpose.orientation, PI
/ 2, 'y')
    cpose.position.y -= 0.15
    cpose.position.x += 0.25
    move_group.go(cpose, wait=True) # goto rotated
point
    move_group.go(bent_home, wait=True)
    # NOTE: in the home position, -x is forwards and y is to
    the right
    # define zero position
    # the following are some helpful quaternion math functions

    # this function returns the mirror of the provided joints
    about the home position
    def mirror_joints(j):
        new_j = j.copy()
        for i in range(len(j)): if
            bent_home[i] == 0:
                new_j[i] = -new_j[i]
            else:

```

```

        pass # do nothing, don't have to mirror return

    new_j

def quat_mult(quatA, quatB):
    newquat = quatA
    newquat.x = (quatA.y * quatB.x) + (quatA.x * quatB.y) +
    (quatA.z * quatB.w) - (quatA.w * quatB.z)
    newquat.y = (quatA.x * quatB.z) - (quatA.y * quatB.w) +
    (quatA.z * quatB.x) + (quatA.w * quatB.y)
    newquat.z = (quatA.x * quatB.w) + (quatA.y * quatB.z) -
    (quatA.z * quatB.y) + (quatA.w * quatB.x)
    newquat.w = (quatA.x * quatB.x) - (quatA.y * quatB.y) -
    (quatA.z * quatB.z) - (quatA.w * quatB.w)
    newquat = quat_normalize(newquat)
    return newquat

def quat_rotate(quat, rot, axis='z'): newquat
    = Quaternion()
    if axis=='z':
        newquat.z = sin(rot *
        0.5) newquat.w = cos(rot
        * 0.5)
    elif axis=='x':
        newquat.x = sin(rot *
        0.5) newquat.w = cos(rot
        * 0.5)
else:
    newquat.y = sin(rot * 0.5)
    newquat.w = cos(rot * 0.5)
    # then, normalize the quaternion
    return quat_mult(quat, quat_normalize(newquat))

def quat_normalize(newquat):
    # 1. compute magnitude of quaternion
    mag = (newquat.x ** 2 + newquat.y ** 2 + newquat.z
    ** 2 + newquat.w ** 2) ** 0.5
    # 2. divide
    newquat.x = newquat.x / mag
    newquat.y = newquat.y / mag
    newquat.z = newquat.z / mag
    newquat.w = newquat.w / mag
    return newquat

if name == " main ":
    zero = array(7 *
    [0.0])
    # define the home position
    home = zero.copy()
    home[0] = PI / 2
    # define the bent home position
    bent_home = zero.copy()
    bent_home[5] = -0.4 * PI / 2
    bent_home[4] = PI / 2
    bent_home[3] = -0.8 * PI / 2
    bent_home[0] = 0.6 * PI / 2
main()

```

A.3 URDF Properties and State Files

A.3.1 /hdt_arm.srdf

```

<?xml version="1.0" ?>
<!--This does not replace URDF, and is not an extension of
  URDF.
  This is a format for representing semantic information
  about the robot structure.
  A URDF file must exist for this robot as well, where
  the joints and the links that are referenced
  are defined
-->
<robot name="hdt_arm">
  <!--GROUPS: Representation of a set of joints and
  links. This can be useful for specifying DOF to plan
  for, defining arms, end effectors, etc-->
  <!--LINKS: When a link is specified, the parent joint
  of that link (if it exists) is automatically included-->
  <!--JOINTS: When a joint is specified, the child link
  of that joint (which will always exist) is automatically
  included-->
  <!--CHAINS: When a chain is specified, all the links
  along the chain (including endpoints) are included in the
  group. Additionally, all the joints that are parents to
  included links are also included
  . This means that joints along the chain and the
  parent joint of the base link are included in the
  group-->
  <!--SUBGROUPS: Groups can also be formed by
  referencing to already defined group names-->
  <group name="elevator">
    <chain base_link="pedestal_link" tip_link="
drive2_link" />
  </group>
  <group name="arm">
    <chain base_link="joint3_housing_link" tip_link
="endpoint_link" />
  </group>
  <!--GROUP STATES: Purpose: Define a named state for a
  particular group, in terms of joint values. This
  is useful to define states like 'folded arms'-->
  <group_state name="stow" group="elevator">
    <joint name="joint1" value="-1.5738" />
    <joint name="joint2" value="3.026" />
  </group_state>
  <group_state name="stow" group="arm">
    <joint name="joint3" value="3.114" />
    <joint name="joint4" value="0" />
    <joint name="joint5" value="0" />
    <joint name="joint6" value="3.155" />
    <joint name="joint7" value="0" />
  </group_state>
</robot>

```

```

        <joint name="joint8" value="1.5708" />
        <joint name="joint9" value="0" />
    </group_state>
    <group_state name="zero" group="arm">
        <joint name="joint3" value="0" />
        <joint name="joint4" value="0" />
        <joint name="joint5" value="0" />
        <joint name="joint6" value="0" />
        <joint name="joint7" value="0" />
        <joint name="joint8" value="0" />
        <joint name="joint9" value="0" />
    </group_state>
    <group_state name="zero" group="elevator">
        <joint name="joint1" value="0" />
        <joint name="joint2" value="0" />
    </group_state>
    <group_state name="ready" group="elevator">
        <joint name="joint1" value="0.785" />
        <joint name="joint2" value="0.785" />
    </group_state>
    <group_state name="ready" group="arm">
        <joint name="joint3" value="1.8389" />
        <joint name="joint4" value="0" />
        <joint name="joint5" value="0" />
        <joint name="joint6" value="1.2838" />
        <joint name="joint7" value="1.7001" />
        <joint name="joint8" value="-0.6592" />
        <joint name="joint9" value="-1.6307" />
    </group_state>
    <group_state name="pre-stow" group="elevator">
        <joint name="joint1" value="-1.2708" />
        <joint name="joint2" value="2.8139" />
    </group_state>
    <group_state name="pre-stow" group="arm">
        <joint name="joint3" value="2.814" />
        <joint name="joint4" value="0" />
        <joint name="joint5" value="0" />
        <joint name="joint6" value="2.8549" />
        <joint name="joint7" value="0" />
        <joint name="joint8" value="1.5708" />
        <joint name="joint9" value="0" />
    </group_state>
    <!--END EFFECTOR: Purpose: Represent information about
an end effector.-->
    <end_effector name="end_effector" parent_link=" endpoint_link"
group="arm" parent_group="arm" />
    <!--VIRTUAL JOINT: Purpose: this element defines a virtual
joint between a robot link and an external frame of reference
(considered fixed with respect to the robot)-->
    <virtual_joint name="world_to_base_link" type=" fixed"
parent_frame="world" child_link="base_link" /
>

```

```

    <!--DISABLE COLLISIONS: By default it is assumed that
any link of the robot could potentially come into
collision with any other link in the robot. This tag
disables collision checking between a specified pair of
links. -->
    <disable_collisions link1="drive1_link" link2=" drive2_link"
reason="Never" />
    <disable_collisions link1="drive1_link" link2="
joint2_bumper_link" reason="Never" />
    <disable_collisions link1="drive1_link" link2="
joint2_housing_link" reason="Never" />
    <disable_collisions link1="drive1_link" link2="
joint3_bumper_link" reason="Never" />
    <disable_collisions link1="drive1_link" link2="
joint3_housing_link" reason="Never" />
    <disable_collisions link1="drive1_link" link2=" mount_link"
reason="Never" />
    <disable_collisions link1="drive1_link" link2=" pedestal_link"
reason="Adjacent" />
    <disable_collisions link1="drive1_link" link2=" platform_link"
reason="Never" />
    <disable_collisions link1="drive1_link" link2=" tubel_link"
reason="Adjacent" />
    <disable_collisions link1="drive2_link" link2="
joint2_bumper_link" reason="Never" />
    <disable_collisions link1="drive2_link" link2="
joint2_housing_link" reason="Adjacent" />
    <disable_collisions link1="drive2_link" link2="
joint3_bumper_link" reason="Never" />
    <disable_collisions link1="drive2_link" link2="
joint3_housing_link" reason="Adjacent" />
    <disable_collisions link1="drive2_link" link2=" joint3_link"
reason="Never" />
    <disable_collisions link1="drive2_link" link2=" tubel_link"
reason="Never" />
    <disable_collisions link1="joint2_bumper_link"
link2="joint2_housing_link" reason="Adjacent" />
    <disable_collisions link1="joint2_bumper_link"
link2="joint3_bumper_link" reason="Never" />
    <disable_collisions link1="joint2_bumper_link"
link2="joint3_housing_link" reason="Never" />
    <disable_collisions link1="joint2_bumper_link"
link2="joint3_link" reason="Never" />
    <disable_collisions link1="joint2_bumper_link"
link2="joint4_link" reason="Never" />
    <disable_collisions link1="joint2_bumper_link"
link2="mount_link" reason="Never" />
    <disable_collisions link1="joint2_bumper_link"
link2="pedestal_link" reason="Never" />
    <disable_collisions link1="joint2_bumper_link"
link2="platform_link" reason="User" />
    <disable_collisions link1="joint2_bumper_link"
link2="tubel_link" reason="Never" />

```

```

    <disable_collisions link1="joint2_housing_link"
link2="joint3_bumper_link" reason="Never" />
    <disable_collisions link1="joint2_housing_link"
link2="joint3_housing_link" reason="Never" />
    <disable_collisions link1="joint2_housing_link"
link2="joint3_link" reason="Never" />
    <disable_collisions link1="joint2_housing_link"
link2="joint4_link" reason="Never" />
    <disable_collisions link1="joint2_housing_link"
link2="joint6_bumper_link" reason="User" />
    <disable_collisions link1="joint2_housing_link"
link2="mount_link" reason="Never" />
    <disable_collisions link1="joint2_housing_link"
link2="pedestal_link" reason="Never" />
    <disable_collisions link1="joint2_housing_link"
link2="tubel_link" reason="Adjacent" />
    <disable_collisions link1="joint3_bumper_link"
link2="joint3_housing_link" reason="Adjacent" />
    <disable_collisions link1="joint3_bumper_link"
link2="joint3_link" reason="Never" />
    <disable_collisions link1="joint3_bumper_link"
link2="joint6_bumper_link" reason="Never" />
    <disable_collisions link1="joint3_bumper_link"
link2="joint6_housing_link" reason="Never" />
    <disable_collisions link1="joint3_bumper_link"
link2="joint6_link" reason="Never" />
    <disable_collisions link1="joint3_bumper_link"
link2="mount_link" reason="User" />
    <disable_collisions link1="joint3_bumper_link"
link2="tubel_link" reason="Never" />
    <disable_collisions link1="joint3_housing_link"
link2="joint3_link" reason="Adjacent" />
    <disable_collisions link1="joint3_housing_link"
link2="joint5_link" reason="Never" />
    <disable_collisions link1="joint3_housing_link"
link2="joint6_bumper_link" reason="Never" />
    <disable_collisions link1="joint3_housing_link"
link2="joint6_housing_link" reason="Never" />
    <disable_collisions link1="joint3_housing_link"
link2="joint6_link" reason="Never" />
    <disable_collisions link1="joint3_housing_link"
link2="tubel_link" reason="Never" />
    <disable_collisions link1="joint3_link" link2=" joint4_link"
reason="Adjacent" />
    <disable_collisions link1="joint3_link" link2=" joint5_link"
reason="Never" />
    <disable_collisions link1="joint3_link" link2="
joint6_bumper_link" reason="Never" />
    <disable_collisions link1="joint3_link" link2="
joint6_housing_link" reason="Never" />
    <disable_collisions link1="joint3_link" link2=" joint6_link"
reason="Never" />
    <disable_collisions link1="joint3_link" link2=" mount_link"
reason="Never" />

```

```

    <disable_collisions link1="joint4_link" link2=" joint5_link"
reason="Adjacent" />
    <disable_collisions link1="joint4_link" link2="
joint6_bumper_link" reason="Never" />
    <disable_collisions link1="joint4_link" link2="
joint6_housing_link" reason="Never" />
    <disable_collisions link1="joint4_link" link2=" joint6_link"
reason="Never" />
    <disable_collisions link1="joint4_link" link2=" joint7_link"
reason="Never" />
    <disable_collisions link1="joint5_link" link2="
joint6_bumper_link" reason="Never" />
    <disable_collisions link1="joint5_link" link2="
joint6_housing_link" reason="Adjacent" />
    <disable_collisions link1="joint5_link" link2=" joint6_link"
reason="Never" />
    <disable_collisions link1="joint5_link" link2=" joint7_link"
reason="Never" />
    <disable_collisions link1="joint5_link" link2="
joint8_housing_link" reason="Never" />
    <disable_collisions link1="joint6_bumper_link"
link2="joint6_housing_link" reason="Adjacent" />
    <disable_collisions link1="joint6_bumper_link"
link2="joint6_link" reason="Never" />
    <disable_collisions link1="joint6_bumper_link"
link2="joint7_link" reason="Never" />
    <disable_collisions link1="joint6_bumper_link"
link2="joint8_housing_link" reason="Never" />
    <disable_collisions link1="joint6_bumper_link"
link2="joint8_link" reason="Never" />
    <disable_collisions link1="joint6_bumper_link"
link2="joint9_link" reason="Never" />
    <disable_collisions link1="joint6_housing_link"
link2="joint6_link" reason="Adjacent" />
    <disable_collisions link1="joint6_housing_link"
link2="joint7_link" reason="Never" />
    <disable_collisions link1="joint6_housing_link"
link2="joint8_housing_link" reason="Never" />
    <disable_collisions link1="joint6_housing_link"
link2="joint8_link" reason="Never" />
    <disable_collisions link1="joint6_housing_link"
link2="joint9_link" reason="Never" />
    <disable_collisions link1="joint6_link" link2=" joint7_link"
reason="Adjacent" />
    <disable_collisions link1="joint6_link" link2="
joint8_housing_link" reason="Never" />
    <disable_collisions link1="joint6_link" link2=" joint8_link"
reason="Never" />
    <disable_collisions link1="joint6_link" link2=" joint9_link"
reason="Never" />
    <disable_collisions link1="joint7_link" link2="
joint8_housing_link" reason="Adjacent" />
    <disable_collisions link1="joint7_link" link2=" joint8_link"
reason="Never" />

```

```

    <disable_collisions link1="joint8_housing_link"
link2="joint8_link" reason="Adjacent" />
    <disable_collisions link1="joint8_housing_link"
link2="joint9_link" reason="Never" />
    <disable_collisions link1="joint8_link" link2=" joint9_link"
reason="Adjacent" />
    <disable_collisions link1="mount_link" link2=" pedestal_link"
reason="Adjacent" />
    <disable_collisions link1="mount_link" link2=" platform_link"
reason="Adjacent" />
    <disable_collisions link1="mount_link" link2=" tubel_link"
reason="Never" />
    <disable_collisions link1="pedestal_link" link2=" platform_link"
reason="Default" />
    <disable_collisions link1="pedestal_link" link2="
tubel_link" reason="Never" />
</robot>

```

A.3.2 hdt_arm_positionctrl.urdf

```

<?xml version="1.0" encoding="utf-8" ?>
<!--
=====
-->
<!-- | This document was autogenerated by xacro from
      /home/benjamin/hdt2_ws/src/manipulator_desc
      ription/ urdf/hdt_arm.xacro | -->
<!-- | EDITING THIS FILE BY HAND IS NOT RECOMMENDED
      |-->
<!--
=====
-->
<robot
name="hdt_arm">
  <!--
material
s -->
    <material name="drive_color">
        <color rgba="0.78431372549 0.78431372549
0.78431372549 1.0" />
    </material>
    <material name="link_color">
        <color rgba="0.690196078431 0.666666666667
0.666666666667 1.0" />
    </material>
    <!-- origin of this model is the base
link, it contains nothing -->
    <link name="base_link" />
    <joint name="platform" type="fixed">
        <origin rpy="0 0 0" xyz="0 0 1" />
        <parent link="base_link" />
        <child link="platform_link" />

```

```

    </joint>
    <link name="platform_link">
      <collision>
        <origin rpy="0 0 0" xyz="0.75 0 -0.5" />
        <geometry>
          <!--<mesh filename="package://${
package_name}/meshes/PENDING.STL"/> -->
          <box size="3 2 1" />
        </geometry>
      </collision>
    </visual>
      <origin rpy="0 0 0" xyz="0.75 0 -0.5" />
      <geometry>
        <!--<mesh filename="package://${
package_name}/meshes/PENDING.STL"/> -->
        <box size="3 2 1" />
      </geometry>
      <material name="link_color" />
    </visual>

    <inertial>

    <mass value="0.1" />
    <inertia ixx="0.03" iyy="0.03" izz="0.03"

    ixy="0.0" ixz="0.0" iyz="0.0" />
    </inertial>

    </link>
    <joint name="mount_joint" type="fixed">
      <origin rpy="1.57079632679 0 0" xyz="0 0 0.0835

" />
      <parent link="platform_link" />
      <child link="mount_link" />

    </joint>
    <link name="mount_link">
      <collision>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
          <mesh filename="package://
manipulator_description/meshes/Mount Plate.STL" />
          <!--<box size="${mount_x} ${mount_y} ${
mount_z}"/>-->
        </geometry>
      </collision>
    </visual>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/Mount Plate.STL" />

```

```

                                <!--<box size="${mount_x} ${mount_y} ${
mount_z}"/>-->
                                </geometry>
                                <material name="link_color" />
                                </visual>
</inertial>
<mass value="0.1" />
<inertia ixx="0.03" iyy="0.03" izz="0.03"
ixy="0.0" ixz="0.0" iyz="0.0" />
</inertial>
</link>
<!-- begining of manipulator arm model -->
<!-- pedestal -->
<joint name="pedestal_joint" type="fixed">
    <origin rpy="0 0 0" xyz="0 0 0" />
    <parent link="mount_link" />
    <child link="pedestal_link" />
</joint>
<link name="pedestal_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H035182 - SHAREABLE -
PEDESTAL ASSY, MK4-2.STL" />
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H035182 - SHAREABLE -
PEDESTAL ASSY, MK4-2.STL" />
        </geometry>
        <material name="link_color" />
    </visual>
</link>
<inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.025" />
<inertia ixx="0.002384" ixy="-0.000042" ixz

="0.000017" iyy="0.001955" iyz="-0.000651" izz=" 0.003190"
/>
    </inertial>

</link>
<!-- drive 1 -->
<joint name="joint1" type="revolute">
    <axis xyz="0 0 1" />
    <limit effort="500" lower="-6.28318530718"
upper="6.28318530718" velocity="0.5" />
<origin rpy="0 0 1.57079632679" xyz="0 0 0.0835

```

```

" />

<parent link="pedestal_link" />
<child link="drive1_link" />
<dynamics damping="0.7" friction="0.0" />

</joint>
<link name="drive1_link">
  <collision>
    <origin rpy="0 0 0" xyz="0 0 0.025" />
    <geometry>
      <cylinder length="0.13" radius="0.0475"

/>
      </geometry>
    </collision>
    <visual>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE - ST DRIVE -
NO TS, MK4-2.STL" />
      </geometry>
      <material name="drive_color" />
    </visual>
  <inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="2.488" />
<inertia ixx="0.003102" ixy="0.000003" ixz=

"-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
/>
    </inertial>

  </link>
  <!-- tube 1 -->
  <joint name="tube1_joint" type="fixed">
    <origin rpy="0 0 0" xyz="0.5993 0 0" />
    <parent link="drive1_link" />
    <child link="tube1_link" />
  </joint>
  <link name="tube1_link">
    <collision>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/H034947 - SHAREABLE -
LINK ASSY, PAINTED, ELEVATOR, PROXIMAL.STL" />
      </geometry>
    </collision>
    <visual>

```

```

        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034947 - SHAREABLE -
LINK ASSY, PAINTED, ELEVATOR, PROXIMAL.STL" />
        </geometry>
        <material name="link_color" />
    </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.4" />
<inertia ixx="0.001003" ixy="0.000006" ixz=

"0.000024" iyy="0.258523" iyz="0.000000" izz=" 0.258523"
/>
    </inertial>

</link>
<!-- joint 2 housing -->
<joint name="joint2_housing" type="fixed">
    <origin rpy="3.14159265359 3.14159265359 0" xyz
="0.5993 0 0" />
    <parent link="tube1_link" />
    <child link="joint2_housing_link" />
</joint>
<link name="joint2_housing_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0.025" />
        <geometry>
            <cylinder length="0.13" radius="0.0475"

/>
            </geometry>
        </collision>
        <visual>
            <origin rpy="0 0 0" xyz="0 0 0" />
            <geometry>
                <mesh filename="package://

manipulator_description/meshes/H022069 - SHAREABLE - ST
DRIVE - NO TS, MK4-2.STL" />
            </geometry>
            <material name="drive_color"
/>
        </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="2.488" />
<inertia ixx="0.003102" ixy="0.000003" ixz=

```

```

"-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
/>
</inertial>

</link>
<!-- bumper -->
<joint name="joint2_bumper" type="fixed">
  <origin rpy="0 1.57079632679 0" xyz="0 0 0.06"

/>
  <parent link="joint2_housing_link" />
  <child link="joint2_bumper_link" />

</joint>
<link name="joint2_bumper_link">
  <collision>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
    </geometry>
  </collision>
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
    </geometry>
    <material name="link_color" />
  </visual>
</link>

<inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="0.144" />
<inertia ixx="0.000394" ixy="-0.000146" ixz

="0.000000" iyy="0.000160" iyz="0.000005" izz=" 0.000461"
/>
</inertial>
</link>
<!-- drive 2 -->
<joint name="joint2" type="revolute">
  <axis xyz="0 0 1" />
  <!-- position limits to match parameter settings
on actuator -->
  <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
  <origin rpy="0 0 3.14159265359" xyz="0 0
-0.0854" />
  <parent link="joint2_housing_link" />
  <child link="drive2_link" />

```

```

        <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="drive2_link">
        <collision>
            <origin rpy="0 0 0" xyz="0.5991 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H034948 - SHAREABLE -
LINK ASSY, PAINTED, ELEVATOR, DISTAL.STL" />
            </geometry>
        </collision>
        <visual>
            <origin rpy="0 0 0" xyz="0.5991 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H034948 - SHAREABLE -
LINK ASSY, PAINTED, ELEVATOR, DISTAL.STL" />
            </geometry>
            <material name="link_color" />
        </visual>
    </link>
    <inertial>

    <origin rpy="0 0 0" xyz="0 0 0" />
    <mass value="1.84" />
    <inertia ixx="0.001837" ixy="-0.000035" ixz
        ="-0.008069" iyy="0.415442" iyz="-0.000001" izz=" 0.415093" />
    </inertial>

    </link>
    <!-- joint 3 housing -->
    <joint name="joint3_housing" type="fixed">
    <origin rpy="0 3.14159265359 0" xyz="1.1982 0 0
" />
    <parent link="drive2_link" />
    <child link="joint3_housing_link" />

    </joint>
    <link name="joint3_housing_link">
        <collision>
            <origin rpy="0 0 0" xyz="0 0 0.025" />
            <geometry>
                <cylinder length="0.13" radius="0.0475"

/>
                </geometry>
            </collision>
            <visual>
                <origin rpy="0 0 0" xyz="0 0 0" />
                <geometry>
                    <mesh filename="package://

```

```

manipulator_description/meshes/H022069 - SHAREABLE - ST DRIVE -
  NO TS, MK4-2.STL" />
    </geometry>
    <material name="drive_color" />
  </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="2.488" />
<inertia ixx="0.003102" ixy="0.000003" ixz=

  "-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
/>
  </inertial>

</link>
<!-- bumper -->
<joint name="joint3_bumper" type="fixed">
  <origin rpy="0 1.57079632679 0" xyz="0 0 0.06"

/>
  <parent link="joint3_housing_link" />
  <child link="joint3_bumper_link" />

</joint>
<link name="joint3_bumper_link">
  <collision>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE
REVERSED - RING CLAMP ASSY, BUMPER.STL" />
    </geometry>
  </collision>
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
    </geometry>
    <material name="link_color" />
  </visual>

</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="0.144" />
<inertia ixx="0.000394" ixy="-0.000146" ixz

  ="0.000000" iyy="0.000160" iyz="0.000005" izz=" 0.000461"
/>

```

```

        </inertial>
    </link>
    <!-- joint 3 -->
    <joint name="joint3" type="revolute">
        <axis xyz="1 0 0" />
        <!-- position limits to match parameter settings
on actuator -->
        <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
        <origin rpy="1.57079632679 -1.57079632679 0"
xyz="0 0 -0.0897" />
        <parent link="joint3_housing_link" />
        <child link="joint3_link" />
        <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="joint3_link">
        <collision>
            <origin rpy="0 0 0" xyz="0 0 0.025" />
            <geometry>
                <cylinder length="0.13" radius="0.0475"

/>
            </geometry>
        </collision>
        <visual>
            <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE - ST DRIVE -
NO TS, MK4-2.STL" />
            </geometry>
            <material name="drive_color" />
        </visual>
    </link>
    <inertial>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <mass value="2.488" />
        <inertia ixx="0.003102" ixy="0.000003" ixz=

        "-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
        />
    </inertial>
</link>
<!-- joint 4 -->
<joint name="joint4" type="revolute">
    <axis xyz="1 0 0" />
    <!-- position limits to match parameter settings
on actuator -->
    <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
    <origin rpy="1.57079632679 -1.57079632679 0"
xyz="0 0 -0.0897" />
    <parent link="joint3_link" />

```

```

        <child link="joint4_link" />
        <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="joint4_link">
        <collision>
            <origin rpy="0 0 0" xyz="0 0 0.025" />
            <geometry>
                <cylinder length="0.13" radius="0.0475"

/>
            </geometry>
        </collision>
        <visual>
            <origin rpy="0 0 0" xyz="0 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE - ST DRIVE -
NO TS, MK4-2.STL" />
            </geometry>
            <material name="drive_color"
/>
        </visual>
    </link>
    <inertial>

    <origin rpy="0 0 0" xyz="0 0 0" />
    <mass value="2.488" />
    <inertia ixx="0.003102" ixy="0.000003" ixz=

        "-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
        />
    </inertial>
</link>
<!-- joint 5 -->
<joint name="joint5" type="revolute">
    <axis xyz="0 0 1" />
    <!-- position limits to match parameter settings
on actuator -->
    <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
    <origin rpy="0 0 3.14159265359" xyz="0 0
-0.59785" />
    <parent link="joint4_link" />
    <child link="joint5_link" />
    <dynamics damping="0.7" friction="0.0" />
</joint>
<link name="joint5_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034994 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, PROXIMAL.STL" />

```

```

        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034994 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, PROXIMAL.STL" />
        </geometry>
        <material name="link_color" />
    </visual>
</inertial>
        <origin rpy="0 0 0" xyz="0 0 0"
/>
<mass value="1.69" />
        <inertia ixx="0.299973" ixy="0.000000" ixz=" "-
0.001056" iyy="0.299943" iyz="0.000489" izz=" 0.001230" />
    </inertial>

</link>
<!-- joint 6 housing -->
<joint name="joint6_housing" type="fixed">
    <origin rpy="0 -1.57079632679 1.57079632679"
xyz="0 0 -0.59785" />
    <parent link="joint5_link" />
    <child link="joint6_housing_link" />
</joint>
<link name="joint6_housing_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0.025" />
        <geometry>
            <cylinder length="0.13" radius="0.0475"

/>
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://

manipulator_description/meshes/H022070 - SHAREABLE - MT DRIVE
- NO TS, MK4-2.STL" />
        </geometry>
        <material name="drive_color" />
    </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.514" />
<inertia ixx="0.001757" ixy="0.000003" ixz=

```

```

    "-0.000011" iyy="0.001843" iyz="0.000007" izz=" 0.001210"
  />
    </inertial>

  </link>
  <!-- bumper -->
  <joint name="joint6_bumper" type="fixed">
    <origin rpy="0 1.57079632679 0" xyz="0 0 0.06"

/>
    <parent link="joint6_housing_link" />
    <child link="joint6_bumper_link" />
  </joint>
  <link name="joint6_bumper_link">
    <collision>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
      </geometry>
    </collision>
    <visual>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
      </geometry>
      <material name="link_color" />
    </visual>

<inertial>

  <origin rpy="0 0 0" xyz="0 0 0" />
  <mass value="0.144" />
  <inertia ixx="0.000394" ixy="-0.000146" ixz

    ="0.000000" iyy="0.000160" iyz="0.000005" izz=" 0.000461"
  />
    </inertial>
  </link>
  <!-- joint 6 -->
  <joint name="joint6" type="revolute">
    <axis xyz="1 0 0" />
    <!-- position limits to match parameter settings
on actuator -->
    <limit effort="180" lower="-6.28" upper="6.28"
velocity="0.785" />
    <origin rpy="3.14159265359 -1.57079632679 0"
xyz="0 0 -0.0897" />

```

```

        <parent link="joint6_housing_link" />
        <child link="joint6_link" />
        <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="joint6_link">
    <collision>
    <origin rpy="0 0 0" xyz="0 0 0.025" />
    <geometry>
        <cylinder length="0.13" radius="0.0475"
    />
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H022070 - SHAREABLE - MT DRIVE -
NO TS, MK4-2.STL" />
        </geometry>
        <material name="drive_color" />
    </visual>
    <inertial>

    <origin rpy="0 0 0" xyz="0 0 0" />
    <mass value="1.514" />
    <inertia ixx="0.001757" ixy="0.000003" ixz=

        "-0.000011" iyy="0.001843" iyz="0.000007" izz=" 0.001210"
    />
        </inertial>
    </link>
    <!-- joint 7 -->
    <joint name="joint7" type="revolute">
        <axis xyz="0 0 1" />
        <!-- position limits to match parameter settings
on actuator -->
        <limit effort="180" lower="-6.28" upper="6.28"
velocity="0.785" />
        <origin rpy="0 0 0" xyz="0 0 -0.594" />
        <parent link="joint6_link" />
        <child link="joint7_link" />
        <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="joint7_link">
        <collision>
            <origin rpy="0 0 0" xyz="0 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H034997 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, DISTAL.STL" />
            </geometry>

```

```

    </collision>
    <visual>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/H034997 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, DISTAL.STL" />
      </geometry>
      <material name="link_color" />
    </visual>
  </inertial>

  <origin rpy="0 0 0" xyz="0 0 0" />
  <mass value="1.15" />
  <inertia ixx="0.189427" ixy="-0.000007" ixz

    ="0.000135" iyy="0.189424" iyz="-0.000139" izz=" 0.000916"
  />
    </inertial>

  </link>
  <!-- joint 8 housing -->
  <joint name="joint8_housing" type="fixed">
    <origin rpy="-1.57079632679 -1.57079632679 0"
xyz="0 0 -0.59785" />
    <parent link="joint7_link" />
    <child link="joint8_housing_link" />
  </joint>
  <link name="joint8_housing_link">
    <collision>
      <origin rpy="0 0 0" xyz="0 0 -0.01" />
      <geometry>
        <cylinder length="0.07" radius="0.03" /

    >
      </geometry>
    </collision>
    <visual>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://

manipulator_description/meshes/H032627 - SHAREABLE - A48
DRIVE MODULE - NON TORQUE SENSING.STL" />
      </geometry>
      <material name="drive_color" />
    </visual>

    upper="6.28"

    velocity="0.785" />

    <origin rpy="3.14159265359 -1.57079632679 0"
xyz="0 0 -0.0738" />

```

```

    <parent link="joint8_housing_link" />
    <child link="joint8_link" />
    <dynamics damping="0.7" friction="0.0" />
  </joint>
  <link name="joint8_link">
    <collision>
      <origin rpy="0 0 0" xyz="0 0 -0.01" />
      <geometry>
        <cylinder length="0.07" radius="0.03" /
>
        </geometry>
      </collision>
      <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
          <mesh filename="package://
manipulator_description/meshes/H032627 - SHAREABLE - A48 DRIVE
MODULE - NON TORQUE SENSING.STL" />
        </geometry>
        <material name="drive_color" />
      </visual>
    </inertial>

    <origin rpy="0 0 0" xyz="0 0 0" />
    <mass value="0.367" />
    <inertia ixx="0.000229" ixy="0.000000" ixz=
"0.000000" iyy="0.000234" iyz="0.000000" izz="
0.000112" />
    </inertial>
  </link>
  <!-- joint 9 -->
  <joint name="joint9" type="revolute">
    <axis xyz="0 0 1" />
    <!-- position limits to match parameter settings
on actuator -->
    <limit effort="48" lower="-6.28" upper="6.28"
velocity="0.785" />
    <origin rpy="0 0 3.14159265359" xyz="0 0
-0.0697" />
    <parent link="joint8_link" />
    <child link="joint9_link" />
    <dynamics damping="0.7" friction="0.0" />
  </joint>
  <link name="joint9_link">
    <collision>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/H032018 - SHAREABLE -
A48 PEDESTAL ASSY.STL" />
      </geometry>

```

```

    </collision>
    <visual>
      <origin rpy="0 0 0" xyz="0 0 0" />
      <geometry>
        <mesh filename="package://
manipulator_description/meshes/H032018 - SHAREABLE -
A48 PEDESTAL ASSY.STL" />
      </geometry>
      <material name="link_color" />
    </visual>
  </inertial>

  <origin rpy="0 0 0" xyz="0 0 0" />
  <mass value="0.099" />
  <inertia ixx="0.000073" ixy="-0.000006" ixz
    ="0.000002" iyy="0.000074" iyz="0.000005" izz=" 0.000079"
  />
</inertial>
hardwareInterface>
  </joint>

  <actuator name="joint1_motor">
    <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
    <mechanicalReduction>1</mechanicalReduction>

  >
    </actuator>
  </transmission>
  <transmission name="trans_joint2">
    <type>transmission_interface/SimpleTransmission

  </type>
    <joint name="joint2">
      <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
    </joint>
    <actuator name="joint2_motor">
      <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
      <mechanicalReduction>1</mechanicalReduction>

    >
      </actuator>
    </transmission>
    <transmission name="trans_joint3">
      <type>transmission_interface/SimpleTransmission

  </type>
    <joint name="joint3">

```

```

        <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
</joint>
<actuator name="joint3_motor">
    <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
    <mechanicalReduction>1</mechanicalReduction>

>
    </actuator>
</transmission>
<transmission name="trans_joint4">
    <type>transmission_interface/SimpleTransmission

</type>
    <joint name="joint4">
        <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
    </joint>
    <actuator name="joint4_motor">
        <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
        <mechanicalReduction>1</mechanicalReduction>

>
    </actuator>
</transmission>
<transmission name="trans_joint5">
    <type>transmission_interface/SimpleTransmission

</type>
    <joint name="joint5">
        <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
    </joint>
    <actuator name="joint5_motor">
        <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
        <mechanicalReduction>1</mechanicalReduction>

>
    </actuator>
</transmission>
<transmission name="trans_joint6">
    <type>transmission_interface/SimpleTransmission

</type>
    <joint name="joint6">
        <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
    </joint>

```

```

        <actuator name="joint6_motor">
            <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
<mechanicalReduction>1</mechanicalReduction
>
        </actuator>
    </transmission>
    <transmission name="trans_joint7">
        <type>transmission_interface/SimpleTransmission

</type>
        <joint name="joint7">
            <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
        </joint>
        <actuator name="joint7_motor">
            <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction

>
        </actuator>
    </transmission>
    <transmission name="trans_joint8">
        <type>transmission_interface/SimpleTransmission

</type>
        <joint name="joint8">
            <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
        </joint>
        <actuator name="joint8_motor">
            <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction

>
        </actuator>
    </transmission>
    <transmission name="trans_joint9">
        <type>transmission_interface/SimpleTransmission

</type>
        <joint name="joint9">
            <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
        </joint>
        <actuator name="joint9_motor">
            <hardwareInterface>hardware_interface/
PositionJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction

```

```

>
</actuator>
</transmission>
  <gazebo>
    <plugin name="gazebo_ros_control" filename="
libgazebo_ros_control.so">
      <robotNamespace>/</robotNamespace>
    </plugin>
  </gazebo>
</robot>

```

A.3.3 hdt_arm_effortctrl.urdf

```

<?xml version="1.0" encoding="utf-8" ?>
<!--
=====
-->
<!-- | This document was autogenerated by xacro from
      /home/benjamin/hdt2_ws/src/manipulator_desc
      ription/ urdf/hdt_arm.xacro | -->
<!-- | EDITING THIS FILE BY HAND IS NOT RECOMMENDED
      |-->

<!--
=====
-->
  <robot
name="hdt_arm">
  <!--
material
s -->
    <material name="drive_color">
      <color rgba="0.78431372549 0.78431372549
0.78431372549 1.0" />
    </material>
    <material name="link_color">
      <color rgba="0.690196078431 0.666666666667
0.666666666667 1.0" />
    </material>
    <!-- origin of this model is the base
link, it contains nothing -->
    <link name="base_link" />
    <joint name="platform" type="fixed">
      <origin rpy="0 0 0" xyz="0 0 1" />
      <parent link="base_link" />
      <child link="platform_link" />
    </joint>
    <link name="platform_link">
      <collision>
        <origin rpy="0 0 0" xyz="0.75 0 -0.5" />
      </collision>
    </link>
  </material>
  <!--<mesh filename="package://${
package_name}/meshes/PENDING.STL"/> -->

```

```

        <box size="3 2 1" />
    </geometry>
</collision>
<visual>
    <origin rpy="0 0 0" xyz="0.75 0 -0.5" />
    <geometry>
        <!--<mesh filename="package://{
package_name}/meshes/PENDING.STL"/> -->
        <box size="3 2 1" />
    </geometry>
    <material name="link_color" />
</visual>

<inertial>

<mass value="0.1" />
<inertia ixx="0.03" iyy="0.03" izz="0.03"

ixy="0.0" ixz="0.0" iyz="0.0" />
</inertial>

</link>
<joint name="mount_joint" type="fixed">
    <origin rpy="1.57079632679 0 0" xyz="0 0 0.0835

" />
    <parent link="platform_link" />
    <child link="mount_link" />

</joint>
<link name="mount_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/Mount Plate.STL" />
            <!--<box size="{mount_x} {mount_y} {
mount_z}"/>-->
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/Mount Plate.STL" />
            <!--<box size="{mount_x} {mount_y} {
mount_z}"/>-->
        </geometry>
        <material name="link_color" />
    </visual>

<inertial>

```

```

<mass value="0.1" />
<inertia ixx="0.03" iyy="0.03" izz="0.03"

    ixy="0.0" ixz="0.0" iyz="0.0" />
    </inertial>

</link>
<!-- begining of manipulator arm model -->
<!-- pedestal -->
<joint name="pedestal_joint" type="fixed">
    <origin rpy="0 0 0" xyz="0 0 0" />
    <parent link="mount_link" />
    <child link="pedestal_link" />
</joint>
<link name="pedestal_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H035182 - SHAREABLE -
PEDESTAL ASSY, MK4-2.STL" />
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H035182 - SHAREABLE -
PEDESTAL ASSY, MK4-2.STL" />
        </geometry>
        <material name="link_color" />
    </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.025" />
<inertia ixx="0.002384" ixy="-0.000042" ixz

    ="0.000017" iyy="0.001955" iyz="-0.000651" izz=" 0.003190"
    />
</inertial>
</link>
<!-- drive 1 -->
<joint name="joint1" type="revolute">
    <axis xyz="0 0 1" />
    <limit effort="500" lower="-6.28318530718"
upper="6.28318530718" velocity="0.5" />
    <origin rpy="0 0 1.57079632679" xyz="0 0 0.0835

" />
    <parent link="pedestal_link" />
    <child link="drive1_link" />

```

```

        <dynamics damping="0.7" friction="0.0" />

</joint>
<link name="drive1_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0.025" />
        <geometry>
            <cylinder length="0.13" radius="0.0475"

/>
            </geometry>
        </collision>
        <visual>
            <origin rpy="0 0 0" xyz="0 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE - ST DRIVE -
NO TS, MK4-2.STL" />
            </geometry>
            <material name="drive_color" />
        </visual>
    <inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="2.488" />
<inertia ixx="0.003102" ixy="0.000003" ixz=

"-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
/>
        </inertial>

    </link>
    <!-- tube 1 -->
    <joint name="tubel_joint" type="fixed">
        <origin rpy="0 0 0" xyz="0.5993 0 0" />
        <parent link="drive1_link" />
        <child link="tubel_link" />
    </joint>
    <link name="tubel_link">
        <collision>
            <origin rpy="0 0 0" xyz="0 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H034947 - SHAREABLE -
LINK ASSY, PAINTED, ELEVATOR, PROXIMAL.STL" />
            </geometry>
        </collision>
        <visual>
            <origin rpy="0 0 0" xyz="0 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H034947 - SHAREABLE -

```

```

    LINK ASSY, PAINTED, ELEVATOR, PROXIMAL.STL" />
      </geometry>
      <material name="link_color" />
    </visual>
  </inertial>

  <origin rpy="0 0 0" xyz="0 0 0" />
  <mass value="1.4" />
  <inertia ixx="0.001003" ixy="0.000006" ixz=

    "0.000024" iyy="0.258523" iyz="0.000000" izz=" 0.258523"
  />
    </inertial>

  </link>
  <!-- joint 2 housing -->
  <joint name="joint2_housing" type="fixed">
    <origin rpy="3.14159265359 3.14159265359 0" xyz
    ="0.5993 0 0" />
    <parent link="tube1_link" />
    <child link="joint2_housing_link" />
  </joint>
  <link name="joint2_housing_link">
    <collision>
      <origin rpy="0 0 0" xyz="0 0 0.025" />
      <geometry>
        <cylinder length="0.13" radius="0.0475"

      />
    </collision>
    </geometry>
  </collision>
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0"
    />
  </visual>
  <geometry>
    <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE -
ST DRIVE - NO TS, MK4-2.STL" />
  </geometry>
  <material name="drive_color" />
  </visual>
  </inertial>

  <origin rpy="0 0 0" xyz="0 0 0" />
  <mass value="2.488" />
  <inertia ixx="0.003102" ixy="0.000003" ixz=

    "-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
  />
    </inertial>

  </link>

```

```

    <!-- bumper -->
    <joint name="joint2_bumper" type="fixed">
        <origin rpy="0 1.57079632679 0" xyz="0 0 0.06"

/>
        <parent link="joint2_housing_link" />
        <child link="joint2_bumper_link" />

</joint>
<link name="joint2_bumper_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
        </geometry>
        <material name="link_color" />
    </visual>
<inertial>
        <origin rpy="0 0 0" xyz="0 0 0"
/>

<mass value="0.144" />
        <inertia ixx="0.000394" ixy="-0.000146" ixz
="0.000000" iyy="0.000160" iyz="0.000005" izz="
0.000461" />
    </inertial>

</link>
<!-- drive 2 -->
<joint name="joint2" type="revolute">
    <axis xyz="0 0 1" />
    <!-- position limits to match parameter settings
on actuator -->
    <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
    <origin rpy="0 0 3.14159265359" xyz="0 0
-0.0854" />
    <parent link="joint2_housing_link" />
    <child link="drive2_link" />
    <dynamics damping="0.7" friction="0.0" />
</joint>
<link name="drive2_link">
    <collision>
        <origin rpy="0 0 0" xyz="0.5991 0 0" />

```

```

        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034948 - SHAREABLE -
LINK ASSY, PAINTED, ELEVATOR, DISTAL.STL" />
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0.5991 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034948 - SHAREABLE -
LINK ASSY, PAINTED, ELEVATOR, DISTAL.STL" />
        </geometry>
        <material name="link_color" />
    </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.84" />
<inertia ixx="0.001837" ixy="-0.000035" ixz
    ="-0.008069" iyy="0.415442" iyz="-0.000001" izz="
0.415093" />
</inertial>
</link>
<!-- joint 3 housing -->
<joint name="joint3_housing" type="fixed">
    <origin rpy="0 3.14159265359 0" xyz="1.1982 0 0
" />

<parent link="drive2_link" />
<child link="joint3_housing_link" />

</joint>
<link name="joint3_housing_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0.025" />
        <geometry>
            <cylinder length="0.13" radius="0.0475"

/>
        </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE - ST DRIVE -
NO TS, MK4-2.STL" />
        </geometry>

```

```

        <material name="drive_color" />
    </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="2.488" />
<inertia ixx="0.003102" ixy="0.000003" ixz=

    "-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
/>
    </inertial>

</link>
<!-- bumper -->
<joint name="joint3_bumper" type="fixed">
    <origin rpy="0 1.57079632679 0" xyz="0 0 0.06"

/>
    <parent link="joint3_housing_link" />
    <child link="joint3_bumper_link" />

</joint>
<link name="joint3_bumper_link">
<collision>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
        <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
    </geometry>
    </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE REVERSED
- RING CLAMP ASSY, BUMPER.STL" />
            </geometry>
            <material name="link_color" />
        </visual>

</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="0.144" />
<inertia ixx="0.000394" ixy="-0.000146" ixz

    ="0.000000" iyy="0.000160" iyz="0.000005" izz=" 0.000461"
/>
    </inertial>
</link>
<!-- joint 3 -->

```

```

    <joint name="joint3" type="revolute">
      <axis xyz="1 0 0" />
      <!-- position limits to match parameter settings
on actuator -->
      <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
      <origin rpy="1.57079632679 -1.57079632679 0"
xyz="0 0 -0.0897" />
      <parent link="joint3_housing_link" />
      <child link="joint3_link" />
      <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="joint3_link">
      <collision>
        <origin rpy="0 0 0" xyz="0 0 0.025" />
        <geometry>
<cylinder length="0.13" radius="0.0475"
/>
        </geometry>
      </collision>
      <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
          <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE - ST DRIVE -
NO TS, MK4-2.STL" />
        </geometry>
        <material name="drive_color" />
      </visual>
    <inertial>

    <origin rpy="0 0 0" xyz="0 0 0" />
    <mass value="2.488" />
    <inertia ixx="0.003102" ixy="0.000003" ixz=

    "-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
    />
      </inertial>
    </link>
    <!-- joint 4 -->
    <joint name="joint4" type="revolute">
      <axis xyz="1 0 0" />
      <!-- position limits to match parameter settings
on actuator -->
      <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
      <origin rpy="1.57079632679 -1.57079632679 0"
xyz="0 0 -0.0897" />
      <parent link="joint3_link" />
      <child link="joint4_link" />
      <dynamics damping="0.7" friction="0.0" />
    </joint>

```

```

<link name="joint4_link">
  <collision>
    <origin rpy="0 0 0" xyz="0 0 0.025" />
    <geometry>
      <cylinder length="0.13" radius="0.0475"

/>
    </geometry>
  </collision>
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0"
/>
<geometry>
  <mesh filename="package://
manipulator_description/meshes/H022069 - SHAREABLE -
ST DRIVE - NO TS, MK4-2.STL" />
  </geometry>
  <material name="drive_color" />
</visual>
<inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="2.488" />
<inertia ixx="0.003102" ixy="0.000003" ixz=

  "-0.000025" iyy="0.003339" iyz="0.000009" izz=" 0.002291"
/>
  </inertial>
</link>
<!-- joint 5 -->
<joint name="joint5" type="revolute">
  <axis xyz="0 0 1" />
  <!-- position limits to match parameter settings
on actuator -->
  <limit effort="500" lower="-6.28" upper="6.28"
velocity="0.5" />
  <origin rpy="0 0 3.14159265359" xyz="0 0
-0.59785" />
  <parent link="joint4_link" />
  <child link="joint5_link" />
  <dynamics damping="0.7" friction="0.0" />
</joint>
<link name="joint5_link">
  <collision>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H034994 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, PROXIMAL.STL" />
    </geometry>
  </collision>
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0" />

```

```

        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034994 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, PROXIMAL.STL" />
        </geometry>
        <material name="link_color" />
    </visual>

<inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.69" />
<inertia ixx="0.299973" ixy="0.000000" ixz=

"-0.001056" iyy="0.299943" iyz="0.000489" izz=" 0.001230"
/>
    </inertial>

</link>
<!-- joint 6 housing -->
<joint name="joint6_housing" type="fixed">
    <origin rpy="0 -1.57079632679 1.57079632679"
xyz="0 0 -0.59785" />
    <parent link="joint5_link" />
    <child link="joint6_housing_link" />
</joint>
<link name="joint6_housing_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0.025" />
        <geometry>
            <cylinder length="0.13" radius="0.0475"

/>
            </geometry>
        </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://

manipulator_description/meshes/H022070 - SHAREABLE - MT
DRIVE - NO TS, MK4-2.STL" />
            </geometry>
            <material name="drive_color" />
        </visual>
    </inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.514" />
<inertia ixx="0.001757" ixy="0.000003" ixz=

```

```

    "-0.000011" iyy="0.001843" iyz="0.000007" izz=" 0.001210"
  />
</inertial>
</link>
<!-- bumper -->
<joint name="joint6_bumper" type="fixed">
  <origin rpy="0 1.57079632679 0" xyz="0 0 0.06"

/>
  <parent link="joint6_housing_link" />
  <child link="joint6_bumper_link" />

</joint>
<link name="joint6_bumper_link">
  <collision>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE
REVERSED - RING CLAMP ASSY, BUMPER.STL" />
    </geometry>
  </collision>
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H035357 - SHAREABLE
REVERSED - RING CLAMP ASSY, BUMPER.STL" />
    </geometry>
    <material name="link_color" />
  </visual>
</link>
<inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="0.144" />
<inertia ixx="0.000394" ixy="-0.000146" ixz

    ="0.000000" iyy="0.000160" iyz="0.000005" izz=" 0.000461"
  />
    </inertial>
</link>
<!-- joint 6 -->
<joint name="joint6" type="revolute">
  <axis xyz="1 0 0" />
  <!-- position limits to match parameter settings
on actuator -->
    <limit effort="180" lower="-6.28" upper="6.28"
velocity="0.785" />
  <origin rpy="3.14159265359 -1.57079632679 0"
xyz="0 0 -0.0897" />
  <parent link="joint6_housing_link" />
  <child link="joint6_link" />

```

```

        <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="joint6_link">
        <collision>
            <origin rpy="0 0 0" xyz="0 0 0.025" />
            <geometry>
                <cylinder length="0.13" radius="0.0475"

/>
            </geometry>
        </collision>
        <visual>
            <origin rpy="0 0 0" xyz="0 0 0" />
            <geometry>
                <mesh filename="package://
manipulator_description/meshes/H022070 - SHAREABLE - MT DRIVE -
NO TS, MK4-2.STL" />
            </geometry>
            <material name="drive_color" />
        </visual>
    </link>
    <inertial>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <mass value="1.514" />
        <inertia ixx="0.001757" ixy="0.000003" ixz=
            "-0.000011" iyy="0.001843" iyz="0.000007" izz=" 0.001210"
        />
    </inertial>
</link>
<!-- joint 7 -->
<joint name="joint7" type="revolute">
    <axis xyz="0 0 1" />
    <!-- position limits to match parameter settings
on actuator -->
    <limit effort="180" lower="-6.28" upper="6.28"
velocity="0.785" />
    <origin rpy="0 0 0" xyz="0 0 -0.594" />
    <parent link="joint6_link" />
    <child link="joint7_link" />
    <dynamics damping="0.7" friction="0.0" />
</joint>
<link name="joint7_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034997 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, DISTAL.STL" />
        </geometry>
    </collision>
    <visual>

```

```

        <origin rpy="0 0 0" xyz="0 0 0" />
        <geometry>
            <mesh filename="package://
manipulator_description/meshes/H034997 - SHAREABLE -
LINK ASSY, PAINTED, MANIPULATOR, DISTAL.STL" />
        </geometry>
        <material name="link_color" />
    </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="1.15" />
<inertia ixx="0.189427" ixy="-0.000007" ixz=

="0.000135" iyy="0.189424" iyz="-0.000139" izz=" 0.000916"
/>
    </inertial>

</link>
<!-- joint 8 housing -->
<joint name="joint8_housing" type="fixed">
    <origin rpy="-1.57079632679 -1.57079632679 0"
xyz="0 0 -0.59785" />
    <parent link="joint7_link" />
    <child link="joint8_housing_link" />
</joint>
<link name="joint8_housing_link">
    <collision>
        <origin rpy="0 0 0" xyz="0 0 -0.01" />
        <geometry>
            <cylinder length="0.07" radius="0.03" /
>
            </geometry>
        </collision>
    <visual>
        <origin rpy="0 0 0" xyz="0 0 0" />
    </geometry>
        <mesh filename="package://
manipulator_description/meshes/H032627 - SHAREABLE - A48 DRIVE
MODULE - NON TORQUE SENSING.STL" />
        </geometry>
        <material name="drive_color" />
    </visual>
</inertial>

<origin rpy="0 0 0" xyz="0 0 0" />
<mass value="0.367" />
<inertia ixx="0.000229" ixy="0.000000" ixz=

"0.000000" iyy="0.000234" iyz="0.000000" izz="
0.000112" />

```

```

        </inertial>
    </link>
    <!-- joint 8 -->
    <joint name="joint8" type="revolute">
        <axis xyz="1 0 0" />
        <!-- position limits to match parameter settings
on actuator -->
        <limit effort="48" lower="-6.28" upper="6.28"
velocity="0.785" />
        <origin rpy="3.14159265359 -1.57079632679 0"
xyz="0 0 -0.0738" />
        <parent link="joint8_housing_link" />
        <child link="joint8_link" />
        <dynamics damping="0.7" friction="0.0" />
    </joint>
    <link name="joint8_link">
        <collision>
            <origin rpy="0 0 0" xyz="0 0 -0.01" />
            <geometry>
                <cylinder length="0.07" radius="0.03" /
>
                </geometry>
            </collision>
            <visual>
                <origin rpy="0 0 0" xyz="0 0 0" />
                <geometry>
                    <mesh filename="package://
manipulator_description/meshes/H032627 - SHAREABLE - A48 DRIVE
MODULE - NON TORQUE SENSING.STL" />
                </geometry>
                <material name="drive_color"
/>
            </visual>
        </inertial>

        <origin rpy="0 0 0" xyz="0 0 0" />
        <mass value="0.367" />
        <inertia ixx="0.000229" ixy="0.000000" ixz=

            "0.000000" iyy="0.000234" iyz="0.000000" izz="
0.000112" />
        </inertial>
    </link>
    <!-- joint 9 -->
    <joint name="joint9" type="revolute">
        <axis xyz="0 0 1" />
        <!-- position limits to match parameter settings
on actuator -->
        <limit effort="48" lower="-6.28" upper="6.28"
velocity="0.785" />
        <origin rpy="0 0 3.14159265359" xyz="0 0

```

```

-0.0697" />
  <parent link="joint8_link" />
  <child link="joint9_link" />
  <dynamics damping="0.7" friction="0.0" />
</joint>
<link name="joint9_link">
  <collision>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H032018 - SHAREABLE -
A48 PEDESTAL ASSY.STL" />
    </geometry>
  </collision>
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0" />
    <geometry>
      <mesh filename="package://
manipulator_description/meshes/H032018 - SHAREABLE -
A48 PEDESTAL ASSY.STL" />
    </geometry>
    <material name="link_color" />
  </visual>
  <inertial>
    <origin rpy="0 0 0" xyz="0 0 0"
/>
  <mass value="0.099" />
    <inertia ixx="0.000073" ixy="-0.000006" ixz
="0.000002" iyy="0.000074" iyz="0.000005" izz=" 0.000079"
/>
  </inertial>

</link>
<!-- endpoint -->
<joint name="joint9_link" type="fixed">
  <origin rpy="0 1.57079632679 0" xyz="0 0 -0.1"

/>
  <parent link="joint9_link" />
  <child link="endpoint_link" />

</joint>
<link name="endpoint_link" />

<transmission name="trans_joint1">
  <type>transmission_interface/SimpleTransmission
</type>
  <joint name="joint1">
    <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
  </joint>
  <actuator name="joint1_motor">

```

```

        <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
        <mechanicalReduction>1</mechanicalReduction>

    >
        </actuator>
    </transmission>
    <transmission name="trans_joint2">
        <type>transmission_interface/SimpleTransmission

    </type>
        <joint name="joint2">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
        </joint>
        <actuator name="joint2_motor">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction>

    >
        </actuator>
    </transmission>
<transmission name="trans_joint3">
<type>transmission_interface/SimpleTransmission
</type>
    <joint name="joint3">
        <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
    </joint>
    <actuator name="joint3_motor">
        <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
        <mechanicalReduction>1</mechanicalReduction>

>
    </actuator>
</transmission>
<transmission name="trans_joint4">
    <type>transmission_interface/SimpleTransmission

</type>
    <joint name="joint4">
        <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
    </joint>
    <actuator name="joint4_motor">
        <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
        <mechanicalReduction>1</mechanicalReduction>

>

```

```

        </actuator>
    </transmission>
    <transmission name="trans_joint5">
        <type>transmission_interface/SimpleTransmission

</type>
        <joint name="joint5">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
        </joint>
        <actuator name="joint5_motor">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction

>
        </actuator>
    </transmission>
    <transmission name="trans_joint6">
        <type>transmission_interface/SimpleTransmission

</type>
        <joint name="joint6">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
        </joint>
        <actuator name="joint6_motor">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction

>
        </actuator>
    </transmission>
    <transmission name="trans_joint7">
        <type>transmission_interface/SimpleTransmission

</type>
        <joint name="joint7">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
        </joint>
        <actuator name="joint7_motor">
            <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction

>
        </actuator>
    </transmission>
    <transmission name="trans_joint8">
        <type>transmission_interface/SimpleTransmission

```

```

</type>
  <joint name="joint8">
    <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
  </joint>
  <actuator name="joint8_motor">
    <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
    <mechanicalReduction>1</mechanicalReduction>

>
    </actuator>
  </transmission>
  <transmission name="trans_joint9">
    <type>transmission_interface/SimpleTransmission

</type>
    <joint name="joint9">
      <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
    </joint>
    <actuator name="joint9_motor">
      <hardwareInterface>hardware_interface/
EffortJointInterface</hardwareInterface>
      <mechanicalReduction>1</mechanicalReduction>

>
    </actuator>
  </transmission>
  <gazebo>
    <plugin name="gazebo_ros_control" filename="

libgazebo_ros_control.so">
      <robotNamespace></robotNamespace>
    </plugin>
  </gazebo>
</robot>

```

A.3.4 hdt_arm_properties.xacro

```

<?xml version="1.0"?>
<robot name="hdt_arm" xmlns:xacro="http://www.ros.org/
wiki/xacro">

  <!-- default inertia - set to non-zero values based on
recommendations from ROS.org
    http://wiki.ros.org/urdf/Tutorials/Adding%20
Physical%20and%20Collision%20Properties%20to%20a%20
URDF%20Model-->
  <xacro:macro name="default_inertia">
    <inertia

```

```

        ixz="0.000001" ixy="0.0" ixz="0.0"
        iyy="0.000001" iyz="0.0"
        izz="0.000001"/>
    </xacro:macro>

    <!-- properties of H032018 - SHAREABLE - A48 PEDESTAL
    ASSY -->
    <xacro:property name="H032018_mass" value="0.099"/>
    <xacro:macro name="H032018_inertia">
        <inertia
            ixz="0.000073" ixy="-0.000006" ixz="0.000002"
            iyy="0.000074" iyz="0.000005" izz="0.000079"/>
    </xacro:macro>

    <!-- properties of H031213 - SHAREABLE - A48 DRIVE
    MODULE - TORQUE SENSING -->
    <xacro:property name="H031213_mass" value="0.367"/>
    <xacro:macro name="H031213_inertia">
        <inertia
            ixz="0.000229" ixy="0.000000" ixz="0.000000" iyy="0.000234"
            iyz="0.000000" izz="0.000112"/>
    </xacro:macro>

    <!-- properties of H032627- SHAREABLE - A48 DRIVE MODULE -
    NON TORQUE SENSING -->
    <xacro:property name="H032627_mass" value="0.367"/>
    <xacro:macro name="H032627_inertia">
        <inertia
            ixz="0.000229" ixy="0.000000" ixz="0.000000" iyy="0.000234"
            iyz="0.000000" izz="0.000112"/>
    </xacro:macro>

    <!-- properties of H032690 - SHAREABLE - PROXIMAL TUBE
    ASSY, AXIAL-AXIAL, 351MM -->
    <xacro:property name="H032690_mass" value="0.195"/>
    <xacro:macro name="H032690_inertia">
        <inertia
            ixz="0.000027" ixy="-0.000001" ixz="0.000000"
            iyy="0.002919" iyz="0.000000" izz="0.002919"/>
    </xacro:macro>

    <!-- properties of H032698 - SHAREABLE - A48 ELBOW BRACKET
    COMPUTER ASSY -->
    <xacro:property name="H032698_mass" value="0.255"/>
    <xacro:macro name="H032698_inertia">
        <inertia
            ixz="0.000126" ixy="0.000006" ixz="0.000049" iyy="0.000606"
            iyz="0.000017" izz="0.000642"/>
    </xacro:macro>

    <!-- properties of H032558 - SHAREABLE - SWAT ARM ELBOW
    CAMERA ASSY -->
    <xacro:property name="H032558_mass" value="0.378"/>
    <xacro:macro name="H032558_inertia">

```

```

        <inertia
            ixx="0.001590" ixy="-0.000259" ixz="-0.000255"
            iyy="0.000997" iyz="0.000606"
            izz="0.001211"/>
        </xacro:macro>

    <!-- properties of H032816 - SHAREABLE - A24 DRIVE
    - TORQUE SENSING - A48 NUT -->
    <xacro:property name="H032816_mass" value="0.240"/>
    <xacro:macro name="H032816_inertia">
        <inertia
            ixx="0.000100" ixy="0.000000" ixz="0.000000" iyy="0.000104"
            iyz="0.000000" izz="0.000068"/>
        </xacro:macro>

    <!-- properties of H032817 - SHAREABLE - A24 DRIVE - EMD -
    A48 NUT -->
    <xacro:property name="H032817_mass" value="0.240"/>
    <xacro:macro name="H032817_inertia">
        <inertia
            ixx="0.000100" ixy="0.000000" ixz="0.000000" iyy="0.000104"
            iyz="0.000000" izz="0.000068"/>
        </xacro:macro>

    <!-- properties of H032696 - SHAREABLE - DISTAL TUBE ASSY,
    AXIAL-RADIAL, 270MM -->
    <xacro:property name="H032696_mass" value="0.150"/>
    <xacro:macro name="H032696_inertia">
        <inertia
            ixx="0.001253" ixy="0.000000" ixz="0.000000" iyy="0.001254"
            iyz="0.000001" izz="0.000020"/>
        </xacro:macro>

    <!-- properties of H032455 - SHAREABLE - A24 DRIVE MODULE -
    TORQUE SENSING - EXT -->
    <xacro:property name="H032455_mass" value="0.237"/>
    <xacro:macro name="H032455_inertia">
        <inertia
            ixx="0.000096" ixy="0.000000" ixz="0.000000" iyy="0.000101"
            iyz="0.000000" izz="0.000067"/>
        </xacro:macro>

    <!-- properties of H020840 - SHAREABLE - A24 DRIVE
    MODULE - EMD -->
    <xacro:property name="H020840_mass" value="0.237"/>
    <xacro:macro name="H020840_inertia">
        <inertia
            ixx="0.000096" ixy="0.000000" ixz="0.000000" iyy="0.000101"
            iyz="0.000000" izz="0.000067"/>
        </xacro:macro>

    <!-- properties of H031917 - SHAREABLE - A48 PINCER ASSY, LAND --
    >
    <xacro:property name="H031917_mass" value="0.597"/>

```

```

<xacro:macro name="H031917_inertia">
  <inertia
    ixx="0.000377" ixy="-0.000021" ixz="0.000100" iyy="0.000556"
    iyz="-0.000002" izz="00.000442"/>
</xacro:macro>

<!-- properties of H022927 - SHAREABLE - A35 PINCER FINGER,
LONG, OVERMOLD ASSY -->
<xacro:property name="H022927_mass" value="0.069"/>
<xacro:macro name="H022927_inertia">
  <inertia
    ixx="0.000015" ixy="0.000035" ixz="0.000000" iyy="0.000362"
    iyz="0.000000" izz="0.000375"/>
</xacro:macro>

<!-- properties of H027206 - camera, ethernet -->
<xacro:property name="H027206_mass" value="0.312"/>
<xacro:macro name="H027206_inertia">
  <inertia
    ixx="0.000624" ixy="-0.000052" ixz="-0.000215"
    iyy="0.000709" iyz="0.000013" izz="0.000558"/>
</xacro:macro>

<!-- properties of H034794 - SHAREABLE - PEDESTAL ASSY-Q,
MK4-2 -->
<xacro:property name="H034794_mass" value="0.886"/>
<xacro:macro name="H034794_inertia">
  <inertia
    ixx="0.001652" ixy="-0.000019" ixz="-0.000048"
    iyy="0.002658" iyz="0.000043"
    izz="0.002982"/>
</xacro:macro>

<!-- properties of H022069 - SHAREABLE - ST DRIVE - NO TS,
MK4-2 -->
<xacro:property name="H022069_mass" value="2.488"/>
<xacro:macro name="H022069_inertia">
  <inertia
    ixx="0.003102" ixy="0.000003" ixz="-0.000025"
    iyy="0.003339" iyz="0.000009" izz="0.002291"/>
</xacro:macro>

<!-- properties of H022070 - SHAREABLE - MT DRIVE - NO TS,
MK4-2 -->
<xacro:property name="H022070_mass" value="1.514"/>
<xacro:macro name="H022070_inertia">
  <inertia
    ixx="0.001757" ixy="0.000003" ixz="-0.000011"
    iyy="0.001843" iyz="0.000007" izz="0.001210"/>
</xacro:macro>

<!-- properties of H034947 - SHAREABLE - LINK ASSY, PAINTED,
ELEVATOR, PROXIMAL -->
<xacro:property name="H034947_mass" value="1.400"/>
<xacro:macro name="H034947_inertia">
  <inertia

```

```

        ixz="0.000024" iyy="0.258523"
        iyz="0.000000" izz="0.258523"/>
</xacro:macro>

<!-- properties of H035357 - SHAREABLE REVERSED - RING CLAMP
ASSY, BUMPER -->
<xacro:property name="H035357_mass" value="0.144"/>
<xacro:macro name="H035357_inertia">
    <inertia
        ixz="0.000394" ixy="-0.000146" ixz="0.000000"
        iyy="0.000160" iyz="0.000005" izz="0.000461"/>
</xacro:macro>

<!-- properties of H034994 - SHAREABLE - LINK ASSY, PAINTED,
MANIPULATOR, PROXIMAL -->
<xacro:property name="H034994_mass" value="1.690"/>
<xacro:macro name="H034994_inertia">
    <inertia
        ixz="0.299973" ixy="0.000000" ixz="-0.001056"
        iyy="0.299943" iyz="0.000489" izz="0.001230"/>
</xacro:macro>

<!-- properties of H034997 - SHAREABLE - LINK ASSY,
PAINTED, MANIPULATOR, DISTAL -->
<xacro:property name="H034997_mass" value="1.150"/>
<xacro:macro name="H034997_inertia">
    <inertia
        ixz="0.189427" ixy="-0.000007" ixz="0.000135"
        iyy="0.189424" iyz="-0.000139" izz="0.000916"/>
</xacro:macro>

<!-- properties of H034948 - SHAREABLE - LINK ASSY,
PAINTED, ELEVATOR, DISTAL -->
<xacro:property name="H034948_mass" value="1.840"/>
<xacro:macro name="H034948_inertia">
    <inertia
        ixz="0.001837" ixy="-0.000035" ixz="-0.008069"
        iyy="0.415442" iyz="-0.000001" izz="0.415093"/>
</xacro:macro>

<!-- properties of H035182 - SHAREABLE - PEDESTAL ASSY,
MK4-2 -->
<xacro:property name="H035182_mass" value="1.025"/>
<xacro:macro name="H035182_inertia">
    <inertia
        ixz="0.002384" ixy="-0.000042" ixz="0.000017"
        iyy="0.001955" iyz="-0.000651" izz="0.003190"/>
</xacro:macro>
</robot>

```

A.4 Launch Files

A.4.1 hdt_arm_bringup_1.launch

```
<?xml version="1.0"?>
```

```

<launch>
  <arg name="debug" default="false" />
  <arg name="gazebo" default="false" />
  <arg name="simulation" default="true" />
  <arg name="controller_type" default="ds4" />
  <arg name="velocity_mode" default="false" />

  <!-- rqt console for filtering ROS logging messages by
  node
    <node name="rqt_console" pkg="rqt_console" type
  ="rqt_console"/>
  -->

    <!-- load robot description -->
    <param name="robot_description" textfile="$(find
  manipulator_revamped_moveit)/urdf/revamped.urdf" />

    <!-- load the hardware configuration -->
    <rosparam command="load" file="$(find
  manipulator_description)/config/hardware.yaml" />

    <!-- load the controller configuration -->
    <rosparam command="load" file="$(find
  manipulator_description)/config/controllers.yaml" />

    <!-- load cycle positions using in factory
  acceptance testing -->
    <rosparam command="load" file="$(find
  manipulator_description)/config/cycle_poses.yaml" if
  ="$(arg debug)"/>

    <!--load parameter file mapping joystick axis to
  joints -->
    <!--also configures button mapping for start/
  stop control, etc. -->
    <rosparam command="load" file="$(eval find('
  manipulator_description') + '/config/joystick_map_'
  + controller_type + '.yaml')" />

    <!-- load the serial nodes -->
    <node name="serial_node" pkg="roserial_python"
  type="serial_node.py" respawn="true" output="screen
  " args="tcp 11411" unless="$(arg simulation)"/>
    <node name="roserial_message_info" pkg=" roserial_python"
  type="message_info_service.py" unless="$(arg simulation)"/>

    <!-- load the hardware interface -->
    <node name="adroit_control" pkg=" hdt_adroit_control"
  type="adroit_control" respawn=" false" output="screen">
      <param name="robot_name" value="hdt_arm" />
      <param name="fake_execution" value="$(arg simulation)"
  />

      <param name="hdt_telem_enable" value="$(arg
  debug)" />

```

```

</node>

<!-- load the actuator debug interface and rqt GUI for
interacting with it-->
<node name="adroit_debug" pkg="hdt_adroit_debug"
type="hdt_adroit_console" respawn="false" output="screen"
if="$(arg debug)"/>

<node name="rqt_gui" pkg="rqt_gui" type="rqt_gui"
args="--perspective-file $(find
hdt_adroit_debug)/config/debug_services.perspective"
respawn="false" output="screen" if="$(arg debug)"/>

<!-- load and start controllers for trajectory
control-->
<node name="controller_manager_spawn" pkg="
controller_manager" type="controller_manager"
respawn="false" output="screen" args="spawn
/hdt_arm/joint_state_controller
/hdt_arm/arm_controller
/hdt_arm/elevator_controller" />

<!-- load but do not start controllers for joint
by joint control-->
<node name="controller_manager_load" pkg="
controller_manager" type="controller_manager"
respawn="false" output="screen" args="load
/hdt_arm/joint1_position_controller
/hdt_arm/joint2_position_controller
/hdt_arm/joint3_position_controller
/hdt_arm/joint4_position_controller
/hdt_arm/joint5_position_controller
/hdt_arm/joint6_position_controller
/hdt_arm/joint7_position_controller
/hdt_arm/joint8_position_controller
/hdt_arm/joint9_position_controller" />
<node name="controller_manager_load_vel" pkg="
controller_manager" type="controller_manager"
respawn="false" output="screen"
if="$(arg velocity_mode)" args="load
/hdt_arm/joint1_velocity_controller
/hdt_arm/joint2_velocity_controller
/hdt_arm/joint3_velocity_controller
/hdt_arm/joint4_velocity_controller
/hdt_arm/joint5_velocity_controller
/hdt_arm/joint6_velocity_controller
/hdt_arm/joint7_velocity_controller
/hdt_arm/joint8_velocity_controller
/hdt_arm/joint9_velocity_controller"/>

<node name="rqt_multiplot" pkg="rqt_multiplot"
type="rqt_multiplot" respawn="false" output="screen"
args="--force-discover --multiplot-config $(find
manipulator_bringup)/support_files/multiplot.xml" if

```

```

="${arg debug}">
  </node>

  <!-- start rviz -->
  <include file="$(find manipulator_moveit)/
launch/moveit_rviz.launch">
    <arg name="config" value="true"/>
    <arg name="debug" value="${arg debug}"/>
  </include>

  <!-- shell script to wait for /joint_states topic
-->
  <node name="delay_script" pkg="manipulator_bringup"
type="delay_script.sh" args="$( arg controller_type)"
output="screen"/>
</launch>

```

A.4.2 spawn_model.launch

```

<launch>
  <!-- send urdf to param server -->
  <arg name="spawn_srdf" default="false"/>
  <param name="robot_description" textfile="$(find
hdt_nav)/urdf/hdt_arm_positionctrl.urdf" />

  <!-- push table_description to factory and spawn robot
in gazebo -->
  <node name="spawn_hdt" pkg="gazebo_ros" type="
spawn_model" args="-urdf -param robot_description - model
hdt_arm" respawn="false" output="screen" />

  <param name="robot_semantic_description" textfile="
$(find hdt_nav)/config/hdt_arm.srdf" if="${arg
spawn_srdf}" />
</launch>

```

A.4.3 hdt_nav_node.launch

```

<launch>
  <node name="launched_node" pkg="hdt_nav" type=" hdt_nav_node"
args="" respawn="false"/>
</launch>

```

Abbreviations

API	Application programming interface
CHOMP	Covariant Hamiltonian Optimization for Motion Planning
DoF	Degrees of freedom
ERDC	Engineer Research and Development Center
IK	Inverse kinematics
OMPL	Open Motion Planning Library
ROS	Robot Operating System
SDF	Signed distance fields
SLAM	Simultaneous localization and mapping
SRDF	Semantic Robot Description File
STOMP	Stochastic Trajectory Optimization for Motion Planning
URDF	Universal Robot Description File
USACE	US Army Corps of Engineers

REPORT DOCUMENTATION PAGE

1. REPORT DATE November 2023		2. REPORT TYPE Final Technical Report		3. DATES COVERED <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; vertical-align: top;"> START DATE FY19 </td> <td style="width: 50%; vertical-align: top;"> END DATE FY22 </td> </tr> </table>		START DATE FY19	END DATE FY22	
START DATE FY19	END DATE FY22							
4. TITLE AND SUBTITLE Integrating MOVEit Motion Constraints on a Novel Robotic Manipulator								
5a. CONTRACT NUMBER		5b. GRANT NUMBER		5c. PROGRAM ELEMENT 29785D				
5d. PROJECT NUMBER 497149		5e. TASK NUMBER 8		5f. WORK UNIT NUMBER				
6. AUTHOR(S) Benjamin Christie, Jordan Klein, Anton Netchaev, and Garry Glaspell								
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) US Army Engineer Research and Development Center (ERDC) Information Technology Laboratory (ITL) 3909 Halls Ferry Road Vicksburg, MS 39180-6199 US Army Engineer Research and Development Center (ERDC) Geospatial Research Laboratory (GRL) 7701 Telegraph Road Alexandria, VA 22315-3864				8. PERFORMING ORGANIZATION REPORT NUMBER ERDC TR-23-23				
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Headquarters, US Army Corps of Engineers Washington, DC 20314-1000			10. SPONSOR/MONITOR'S ACRONYM(S) HQUSACE		11. SPONSOR/MONITOR'S REPORT NUMBER(S)			
12. DISTRIBUTION/AVAILABILITY STATEMENT DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.								
13. SUPPLEMENTARY NOTES								
14. ABSTRACT MOVEit, a widely used Robot Operating System framework, plans composite tasks, where the high-level sequence of actions is fixed and known in advance. However, these tasks need to be tailored and adapted to the environmental context. This framework uses custom trajectory planners, known as controllers, to solve goals that are fully defined within the configuration space. Libraries, such as the Open Motion Planning Library, provide a collection of motion planners that can solve task-space goals. An exact spatial and joint replication of the robotic manipulator's mechanics, typically Universal Robot Description Format and Semantic Robot Description Format files, is required. Common arms such as the Panda-Manipulator and OpenMANIPULATOR-X provide these files in their respective public repositories, but custom arms require significant modification or even a complete rewrite of these files.								
15. SUBJECT TERMS Computer programs--Evaluation; Hydraulic structures--inspection; Robots--Control systems; Robots--Kinematics								
16. SECURITY CLASSIFICATION OF: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 33%; vertical-align: top;"> a. REPORT Unclassified </td> <td style="width: 33%; vertical-align: top;"> b. ABSTRACT Unclassified </td> <td style="width: 33%; vertical-align: top;"> c. THIS PAGE Unclassified </td> </tr> </table>			a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified	17. LIMITATION OF ABSTRACT SAR		18. NUMBER OF PAGES 87
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified						
19a. NAME OF RESPONSIBLE PERSON				19b. TELEPHONE NUMBER (include area code)				