LMS

University of Patras

Manual

LABORATORY FOR MANUFACTURING SYSTEMS & AUTOMATION

Deformable material modeling and model-based robot control manual

Laboratory For Manufacturing Systems & Automation
University of Patras

Table of Contents

Introduction

The deformable material modeling and model-based robot control module is a comprehensive toolset designed to facilitate fabric manipulation in robotic applications. This user manual aims to provide detailed instructions and guidance for effectively utilizing the DMC Module. Whether you are an engineer, researcher, or developer, this manual equips you with the essential knowledge required to make the most of this software package.

Module purpose

The primary objective of the module is to offer a versatile platform for simulating fabric manipulation scenarios. It consists of multiple sub-modules, each tailored to specific tasks within the domain of fabric manipulation. These sub-modules include Fabric Simulation, Robotic Planning, and Human Tracking. By integrating these modules, users can streamline the process of simulating and planning fabric manipulation tasks, ultimately enhancing their productivity and efficiency.

Sub-Modules Overview

Fabric Simulation

This sub-module focuses on fabric behavior and simulation. Users can create and configure fabric models, adjusting various parameters to mimic real-world scenarios. The fabric simulation sub-module provides a realistic virtual environment for testing and fine-tuning robotic manipulation tasks involving fabrics.

Robotic Planning

The Robotic Planning sub-module complements the fabric simulation by enabling users to design and optimize robotic manipulation strategies. It facilitates the importation of robot models, definition of kinematics, setting up grasping points, and generating precise robotic plans. This module is essential for planning and executing fabric manipulation tasks with precision.

Human Tracking

The Human Tracking sub-module introduces the capability to monitor and interact with human operators involved in fabric manipulation scenarios. By integrating human tracking data, the coordination between robots and human operators is enhanced, ensuring safe and efficient collaboration in fabric manipulation tasks.

In the following sections of this manual, we will dive into each sub-module in detail, providing step-by-step instructions, tips, and best practices for utilizing the module.

System Requirements

Hardware

For optimal performance, the following are recommended:

- Multi-core processor, preferably with at least 8 cores
- 16GB RAM
- NVIDIA RTX GPU with at least 8GB of VRAM
- 50GB of free disk space

Software

- Ubuntu 20.04
- ROS Noetic
- Nvidia CUDA Toolkit

Getting Started

This section will guide you through the initial steps to launch the module and provide an overview of the user interface. Please ensure you've met the system requirements detailed in Section 2 before proceeding.

Launching Module

Follow the following steps to launch the module. Terminal commands may vary slightly, depending on your system configuration.

1. Activate your ROS environment: source /opt/ros/noetic/setup.bash
2. Navigate to the module directory: cd <directory_path>
3. Launch module: roslaunch merging default.launch

User Interface

Upon successfully launching the module, you will gain access to a comprehensive user interface designed to facilitate your fabric manipulation and robotic planning tasks. This user interface is primarily centered around RViz, a powerful visualization tool that provides an interactive 3D environment for your work.

The central component of the user interface is the RViz main display window. This window serves as the primary canvas for visualizing and interacting with your fabric simulations, robot models, and collaborative co-manipulation scenarios. You can

manipulate the view within this window to examine your simulations and plans from different perspectives.

On the left side of the RViz window, you will find a set of panels, each dedicated to a specific submodule. These panels include the following:

- Fabric Simulation Panel: This panel is the control of the fabric simulation process. Here, you can start, stop, and reset the fabric simulations.

- Robotic Planning Panel: The Robotic Planning Panel is a crucial tool for defining and managing your robotic planning tasks. Within this panel, you can control and orchestrate all aspects of the planning module.

- Toolbar: The toolbar located at the top of the RViz window provides quick access to various functionalities and tools provided by RViz. Additionally, there is a custom tool dedicated to grasping point manipulation for the simulation model of the fabrics.

Fabric Simulation Module

The Fabric Simulation module constitutes an integral element of the framework. Its primary function lies in augmenting the cognitive abilities of robotic systems concerning the complex dynamics of fabric manipulation. This enhancement is realized through a sophisticated simulation of fabric behavior that can operate in real-time alongside live human-robot co-manipulation activities, or in a fully simulated, offline environment.

The module is designed to emulate, with high fidelity, how fabrics respond to a multitude of external forces and manipulations. This capability is vital for improving the system's understanding of various fabric characteristics. In addition, the Fabric Simulation module collaborates closely with the system's digital twin, establishing an uninterrupted link between physical robotic operations and their virtual representations. This synergy introduces robot-specific interfaces into the simulation, thereby augmenting the module's overall modeling efficacy within the broader framework.

In summary, the Fabric Simulation module functions as a critical intermediary between actual fabric materials and their digital representations, facilitating advanced robotic cognition. It enables robotic systems to better anticipate, adapt to, and manage the intricate behaviors exhibited by fabrics during manipulation tasks. Leveraging real-time simulation capabilities, the module enhances the performance and dexterity of robotic agents, aligning them more closely with human-like interaction and understanding of fabric manipulation.

Operating the Simulator

The simulation tool operates analogously to comparable simulation software, offering multiple user interfaces that are seamlessly integrated with existing ROS utilities. To initiate the test environment, execute the command roslaunch clothsim sim.launch, which will load a test fabric along with a predefined collision scenario. By default, the simulation initializes in a halted state.

The simulation's state can be controlled via the ROS service exposed at /clothsim/SetState, which utilizes the std_srvs::SetBool type. The boolean data within this service determines the simulation's active or paused state. To halt the simulation, simply send a "false" value to this service endpoint.

Resetting the simulation is possible via the /clothsim/Reset service that employs a std_srvs::Trigger type. It's critical to recognize that upon resetting, the simulation reverts to its last known state. Hence, a running simulation will automatically resume post-reset, while a paused simulation will restart in a halted state.

Beyond the ROS-based interfaces, the tool's graphical user interface (GUI) delivers all necessary functionality through an RViz plugin. After integrating this panel into the RViz interface, users can execute all requisite commands adjacent to the simulation's visual display.

RViz, the standard visualization tool within the ROS ecosystem, is used as a visualization environment. Visualization is carried out through pre-configured ROS messages, and visual settings can be customized directly within the RViz interface. A default RViz visualization setup is provided with the tool and is open for user modifications.

Defining Fabrics

The preliminary step in utilizing the Fabric Simulation module involves defining the properties of the fabric material to be simulated. This entails two essential components: a yaml configuration file and a bitmap image serving as the image mask of the target fabric.

The .yaml configuration file houses a set of mandatory parameters that guide the simulation model. These parameters fall into two broad categories, namely Dimensional Parameters and Behavioral Parameters. For the purpose of this section, we focus on Dimensional Parameters:

- SizeX and SizeY: These are double-precision floating-point values that denote the dimensions of the simulated material along the X and Y axes, measured in meters. The default value for both parameters is 1.0 meters.

- X, Y, and Z: These parameters define the position of the center of the simulated fabric in a three-dimensional Cartesian coordinate system. Each is a double-precision floating-point value, set to 0.0 meters by default.

- ParticlesX and ParticlesY: These are unsigned integers indicating the number of particles to be used along the X and Y axes in the simulation, respectively. The default value for each is 20.

- UseResolution and Resolution: The UseResolution flag, when set, enables the tool to automatically calculate the required particle numbers based on the material size and a given resolution value. Resolution is an unsigned integer specifying the density of particles, measured in particles per meter. The default value for Resolution is 20.

To complete the fabric definition, a bitmap image must be supplied as an image mask. The pixel dimensions of this image must correspond with the particle numbers specified in the .yaml configuration file. Each pixel in the image is binary, either black or white, indicating whether the corresponding area should be part of the simulation (white) or not (black).

Once you've created the yaml configuration file and image mask, place both items in a designated folder. To make these accessible by the Fabric Simulation module, load the .yaml file into the ROS Parameter Server. Ensure that it is stored within the same namespace to facilitate proper communication between the configuration and the simulation tool.

Configuring fabric behavior and simulation

The configuration of fabric behavior and the underlying simulation involve a multi-dimensional parameter space. The intention behind exposing these parameters is to offer users a significant degree of control over the simulation behavior, thereby enhancing the adaptability and efficacy of the tool. The parameters can be grouped into various categories, each serving a specific functional role within the broader system.

Name	Type	Default	Description
Simulation Parameters			
Enabled	Boolean	TRUE	Controls the initial state of the simulation.
dt	Double	0.005	The progressing time step of the simulation.
UsePause	Boolean	TRUE	Indicates whether to pause the simulation when possible.
Window	Unsigned Integer	200	The size of the buffer for calculating simulation convergence.
Threshold	Double	0.001	The threshold for simulation convergence.
Dimensional Parameters			
SizeX	Double	1	The size of the simulated material in the X axis.
SizeY	Double	1	The size of the simulated material in the Y axis.
X	Double	0	The position of the center of the simulated material in the X axis.
Y	Double	0	The position of the center of the simulated material in the Y axis.
Z	Double	0	The position of the center of the simulated material in the Z axis.
ParticlesX	Unsigned Integer	20	The number of particles to use in the X axis.
ParticlesY	Unsigned Integer	20	The number of particles to use in the Y axis.
UseResolution	Boolean	FALSE	If set, tool will use resolution value for particle grid calculation.
Resolution	Unsigned Integer	20	The resolution of particles.
Level	Double	-0.1	The height of an imaginary 2D collision ground plane.
Physical Parameters			
Weight	Double	1	The total weight of the simulated object.
Damping	Double	0.95	Damping coefficient for particle motion.
ConstraintIters	Unsigned Integer	0	Number of constraint solving iterations.
StructuralStrength	Double	0.7	Percentage correction for unstable structural springs.
ShearStrength	Double	0.5	Percentage correction for unstable shear springs.
FlexionStrength	Double	0.6	Percentage correction for unstable flexion springs.
TetherStrength	Double	0.9	Percentage correction for unstable tethering springs.
LengthFactor	Double	1.05	Percentage of spring length for instability consideration.
k_str	Double	1	Structural spring constant.
c_str	Double	0	Damping coefficient of structural damper.
p_str	Double	1	Preload factor of structural springs.
f_str	Double	20	Maximum force structural springs can produce.
k_shr	Double	1	Shear spring constant.
c_shr	Double	0	Damping coefficient of shear damper.
p_shr	Double	1	Preload factor of shear springs.
f_shr	Double	20	Maximum force shear springs can produce.
k_flx	Double	1	Flexion spring constant.
c_flx	Double	0	Damping coefficient of flexion damper.
p_flx	Double	1	Preload factor of flexion springs.
f_flx	Double	20	Maximum force flexion springs can produce.
Interaction Parameters			
CornerGPs	Boolean	TRUE	Automatically create grasping points.
CornerGP_Radius	Double	0.04	Radius of the sphere around corner grasping points.
GP0	Boolean	TRUE	Enable grasping point number 0.
GP1	Boolean	TRUE	Enable grasping point number 1.
GP2	Boolean	TRUE	Enable grasping point number 2.
GP3	Boolean	TRUE	Enable grasping point number 3.
UseJoy	Boolean	FALSE	Enable external control through ROS topics.
UseTF	Boolean	TRUE	Enable external control through ROS TF links.
Visual Parameters			
VisualMarkers	Boolean	TRUE	Automatically bind interactive markers to grasping points.
GP0_Visual	Boolean	TRUE	Enable interactive marker for grasping point number 0.
GP1_Visual	Boolean	TRUE	Enable interactive marker for grasping point number 1.
GP2_Visual	Boolean	TRUE	Enable interactive marker for grasping point number 2.
GP3_Visual	Boolean	TRUE	Enable interactive marker for grasping point number 3.
GP_SphereScale	Double	0.12	Radius of interactive markers.
GP_AxesScale	Double	0.25	Length of interactive arrows.

Online parameterization is a critical feature of this tool, enabling the modification of the simulation parameters without necessitating a recompile or restart of the system.

- YAML Configuration Files and Parameterized Launch Files: A YAML configuration file, specified during the tool's launch, is read upon initialization. Users can modify this file to customize the default parameters.

- Dynamic Reconfigure: Implemented in the ROS node responsible for managing the simulation, this feature allows dynamic reconfiguration of parameters. It exposes a Python and C++ API for handling reconfiguration events and offers ROS services to interface with other parts of the system.

All the configuration parameters should be placed inside a dedicated folder. The YAML file containing these parameters must then be loaded into ROS's parameter server to ensure accessibility by the simulation tool. It is crucial that the namespace used in ROS matches that used within the YAML configuration file. By following these guidelines, users can achieve a high degree of control over the simulation, allowing for a wide range of experimental setups and observations.

Thus, the configuration of fabric behavior and the simulation is a multifaceted process but one that offers the user an extensive range of controls for fine-tuning the system to meet specific requirements.

Manipulating Fabrics

Manipulating fabrics within the simulation tool is facilitated by a multi-modal interaction interface designed to be versatile and flexible. The simulation tool offers three primary avenues for external control: 1) Through the communication mechanism enabled by Robot Operating System (ROS) messages, 2) By using interactive markers, and 3) By utilizing TF (Transform) attachments directly to robot links. This layered approach allows researchers and developers to not only interact with the simulated material in real-time but also to integrate the simulation seamlessly into broader robotic control and perception frameworks.

Each method for fabric manipulation offers its unique advantages and use-cases. For instance, ROS messages may be the preferred mode when scripting automation routines or when interacting with other ROS-enabled hardware or software modules. Interactive markers provide a more intuitive, GUI-based mechanism for real-time manipulation, ideal for testing and visualization. TF attachments to robot links offer the most integrated method, allowing the simulated fabric to behave as if it were part of an actual robotic manipulator setup, thereby enabling more realistic simulation studies.

Setting Grasping Points

Setting grasping points on the simulated fabric can be conducted in several ways, designed to provide maximum flexibility for different research requirements. The most straightforward approach is through the SetGP service call, which allows precise definition of grasping points through an API. A more interactive option is available through the GPTool plugin for RViz, where particles can be selected

directly within the simulation's visual interface to set up grasping points. Finally, the simulation also supports SetGP in TF attach mode, whereby grasping points can be automatically defined based on proximity to specified TF links within the simulation environment.

In the SetGP service call, the exact coordinates or indices of the desired grasping points can be provided, offering a high level of precision. The RViz GPTool plugin, on the other hand, provides a more intuitive experience by allowing users to click directly on the graphical representation of the fabric. This visual interaction can be particularly useful for rapid prototyping or for users less familiar with the underlying data structures of the simulation. In TF attach mode, the system automatically attaches grasping points to nearby robot links, providing a highly integrated and dynamic approach for advanced robotic manipulation scenarios.

Robotic Planning Module

Robotic planning serves as a critical component in the development and operation of autonomous systems, responsible for generating optimal paths and sequences of actions that a robot must undertake to fulfill a particular objective (e.g., fabric transfer). Robotic planning takes on an even more important role, allowing for real-time simulation and analysis of planned actions prior to their execution in the physical world. The embedded simulation tool offers a powerful interface for validating planning algorithms, observing potential outcomes, and refining the robot's decision-making processes.

This tool will help generate target goals for the supportive robot agents through the use of the handling inputs of the operator. The inputs of the operator are generated through a stereo camera setup that will be described in later section and are abstracted to provide flexibility in terms of compatibility with tracking systems.

Importing Robot Models

The planning architecture is explicitly engineered to operate seamlessly with the MoveIt framework, thereby eliminating the need for additional customization or configuration. All essential parameters and specifications are auto-extracted from the Unified Robot Description Format (URDF) and the Semantic Robot Description Format (SRDF) that define the robotic system's architecture and functionality. By automatically gleaning this critical information, the planning framework simplifies the integration process, thereby expediting the transition from development to deployment.

Defining Robot Kinematics

For the system to perform optimally, it is imperative to correctly configure it to accept a diverse range of command inputs, including both trajectory and jogging commands, for each specified end effector. This configurational requirement ensures the adaptability and flexibility of the system across various manipulation tasks. Moreover, the Robot Operating System (ROS) controllers integrated into the system are designed to seamlessly accept command inputs in an interchangeable fashion. This feature is fundamental for unlocking the full operational capabilities of the planner. Properly configured ROS controllers serve as the linchpin for the comprehensive functionality,

thereby enabling the planner to execute complex, multi-step actions with high precision and reliability.

Setting Up Grasping Points

The initialization of the planning framework incorporates an automated mechanism for grasping point configuration, thus negating the need for manual adjustments post-initialization. This automation is achieved through a real-time integration with the simulation tool, which actively updates the system's configuration geometry to mirror the current state of the grasping points. Consequently, as the operator or supportive robotic agents interact with the fabric in the digital twin, the planner is equipped to dynamically update the grasping point configurations. This not only streamlines the planning process but also adds an additional layer of adaptability and real-time responsiveness to the system.

This real-time synchronization allows both the operator and supportive agents to focus on the task at hand, rather than being burdened with continual manual adjustments of grasping point settings. The planner's ability to automatically detect and update grasping points ensures that the digital twin's configuration is always current, thereby increasing the accuracy and efficiency of the planning process. The result is a more harmonious, real-time interaction between the operator, the robotic agents, and the simulation tool, which in turn optimizes the planning and execution of fabric manipulation tasks.

Defining Task Objectives

The primary objective of this planning framework is to serve as an auxiliary tool that augments the capabilities of the operator. It accomplishes this by computationally deriving supportive robotic actions based on the operator's inputs and subsequently transmitting these calculated actions to the respective controllers for execution. To offer flexibility and adaptability, the planner is designed with a configurable architecture that can be customized to align with specific requirements or constraints. The configuration settings of the planner are easily accessible and modifiable through a designated .yaml file, which houses the relevant parameters.

This .yaml file serves as the central repository for all tunable parameters related to task objectives and planner functionality. It allows for a tailored planning experience by providing the operator with the ability to adjust various operational metrics. Each parameter within the file is documented to offer a comprehensive understanding of its role and impact within the planning process. By furnishing a customizable environment, the planning framework not only accommodates a broad array of use-cases but also empowers the designer to optimize the system according to the specific demands of the task at hand.

Name	Type	Default	Description
Frequency	Double	120	The planning frequency
Group	String	dual_arm_gantry	The MoveIt group
EEs	UInt[]	["right_gripper_", "left_gripper"]	The end effectors of the group
EEGPs	UInt[]	[0 , 1]	Indices for the assignment to the grasping points
InitializeWithVision	Boolean	TRUE	Initialize with vision system on start
VisionGroup	String	left_camera	MoveIt group for the vision system
PlaceGroup	String	dual_arm_gantry	MoveIt group for the placement sequence
Repositioning	Boolean	TRUE	Enable the repositioning based on the vision offsets
Alpha	Double	0.994	Filtering alpha for the handling inputs
EnableAxis	Boolean[]	[TRUE, TRUE]	Boolean flags for each additional axis planning
AxisTopic	String[]	["torso", "rail"]	Topics for controlling the commanded outputs for each axis
AxisAlpha	Double[]	[0.0, 0.0]	Filtering alpha for each axis
AxisDeadband	Double[]	[0.21, 0.25]	Deadband values for each axis
AxisDefault	Double[]	[0.0, 2.5]	Default values for each axis
AxisSpeed	Double[]	[0.5, 0.3]	Maximum speed for each axis
AxisRef	Double[][3]	[[0, -1, 0], [0, 1, 0]]	Reference vector for each axis
AxisLimits	Double[][2]	[[-1.5707, 1.5707], [0, 5]]	Limits for each axis
ScalingFactor	Double	0.98	Scaling factor for accounting of fabric's stretch
Mirroring	Double	0.33	Mirroring registration distance for translation
Threshold	Double	1.6	Planning threshold limit for penalty system
SingularityWeight	Double	2	Penalty for the singularity
DifferenceWeight	Double	0.5	Penatly for the difference minimization
OverextensionWeight	Double	1.3	Penalty for the overextension limit
FOVWeight	Double	0.7	Penalty for the visibility constraint
ProbingEnabled	Boolean	TRUE	Enable the force control of the grippers
CollisionAvoidance	Boolean	TRUE	Enable the collision avoidance and detection system

Human Tracking Module

The aspect of robotic planning gains an added layer of sophistication and dynamism when human operators are actively incorporated into the decision-making loop. The human tracking module of the DMC delves into the critical role of Human Tracking in this context, focusing on how state-of-the-art tracking technologies can seamlessly integrate with our robotic planning system to facilitate more nuanced, responsive, and cooperative manipulation tasks. Utilizing technologies like Stereolab's ZED2 camera and specialized ROS packages, this module will guide you through the hardware requirements, software configurations, and the pivotal utilization of human tracking data for co-manipulative actions.

Hardware and Software Requirements for Human Tracking

In terms of hardware, a key component is a high-fidelity sensor capable of capturing 3D spatial data. In this framework, Stereolab's ZED2 camera serves as an exemplary model, offering robust capabilities for real-time depth sensing and spatial mapping. This camera works in tandem with an adequately powerful computing system to meet the computational demands of real-time 3D data processing. As for software, the system is reliant on the ZED SDK, which acts as the backbone for interfacing the camera with the computational environment. This Software Development Kit (SDK) is further augmented by its dependency on CUDA, thereby necessitating a CUDA-compatible graphics card. These hardware and software elements synergistically contribute to the accuracy and reliability of human tracking, making them indispensable components of the system.

Utilizing Human Tracking Data for Co-Manipulation

For the human tracking module to integrate seamlessly with the robotic planning framework, specific software configurations are essential. The planner subscribes to the ROS topic /zed_human_tracking/skeletons to capture the skeletal data generated by the tracking system. These data points are then used to compute the necessary supportive actions for the robotic agents in real-time. The entire communication between the tracking and planning modules is facilitated through ROS messages, which ensures low-latency, real-time data exchange crucial for accurate and responsive co-manipulation tasks.

Flexibility is another key feature of this system. Although the default configuration utilizes the ZED2 camera and its associated SDK for human tracking, swapping out to other tracking systems like OpenPose or custom-built solutions is straightforward. The only requirement is that the new system should be able to publish tracking data in a ROS message format that aligns with the type expected by the planner. Once this condition is met, the planning system can continue to generate supportive robotic actions based on the tracking data, irrespective of its source. This modularity allows for easy adaptability and future-proofing of the system.