***2R Robot Manipulator Optimization Research Model***

Jude Kamoona, Elijah Williams, Daniel Fierro

* **Motivation** 3
* **Paper 1:** *High-Frequency Nonlinear Model Predictive Control of a Manipulator 3*
* **Paper 2:***STOMP: Stochastic Trajectory Optimization for Motion Planning 4*
* **Paper 3:** *Increasing Efficiency of Optimization based Path Planning 5*
* **Conclusion** *6*

**Motivation**

The purpose of this paper is to document the research methodology and selection process for an optimization control and path planning algorithm for the Open Avenues Micro-Internship. The following papers were selected for review due to their prominence in the optimization of controls in the space at the moment. The papers were chosen from a range of research papers selected by our mentor, Noe Fontana. The reviewed papers include *High-Frequency Nonlinear Model Predictive Control of a Manipulator by* Sébastien Kleff, Avadesh Meduri, Rohan Budhiraja, Nicolas Mansard, and Ludovic Righetti,*STOMP: Stochastic Trajectory Optimization for Motion Planning by* Mrinal Kalakrishnan, Sachin Chitta Evangelos Theodorou, Peter Pastor, and Stefan Schaal*, and**Increasing Efficiency of Optimization based Path Planning**by* Hao Ding, Gunther Reißig, and Olaf Stursberg. The following details the evaluation and conclusions of the reviewed research, and the chosen paper for implementation for a 2R and 6R KUKA iiwa robot.

***High-Frequency Nonlinear Model Predictive Control of a Manipulator***

Sébastien Kleff, Avadesh Meduri, Rohan Budhiraja, Nicolas Mansard, and Ludovic Righetti

**Intro**

This paper explored solutions for implementing Model Predictive Control (MPC) by separating planning and control for open-loop MPC (low-frequency) or using simplified models of dynamics, which are more restrictive, but less adaptable. In this paper the authors (Kleff, Meduri, et al.) explore a third solution, using Differential Dynamic programming (DDP) to translate a formulation of Optimal Control Problem (OCP) into a discrete nonlinear optimization problem.

**Experiment**

The researchers conducted an experiment demonstrating closed-loop Model Predictive Control and open-loop Model Predictive Control, although the latter being more jerky and costly, with less accuracy (see [comparison on KUKA LWR iiwa R820](https://peertube.laas.fr/w/76fcf68d-8509-4257-a3a4-c1d0eca7a8ba)).

The controller which computes torque at 1kHz based on the full dynamics of the model is proposed using multiple-shooting FDDP to solve optimal control problem in a model predictive control with an open loop controller. The controller computes a feedforward torque combined with a joint impedance controller to optimize controls. The paper discusses another experimentation which tests a closed loop controller which updates the feedforward torque with sensory feedback. As expected, the closed loop controller performs better than the open loop controller since we are considering a nonlinear system. The controller being used within the open loop control is a PD controller. Although, computing the gains can be a bit troublesome in the time horizon. The pick and place task essentially means that the end effector will reach certain positions based on predefined times, defined by the length of the cycles for the manipulator.

**Evaluation**

Strengths to highlight about this algorithm is mainly its speed (1 kHz second) which allows running closed-loop online Model Predictive Control (MPC), relying on feasibility-driven Differential Dynamic Programming (DDP) to optimize the discretized objective function. Furthermore Crocoddyl library is tailored for open-source implementation of this control algorithm, has python bindings, and is capable of computing highly dynamic maneuvers within milliseconds.

Although, some hurdles with implementing this algorithm include translating high-level knowledge of MPC and DDP into an incorporated implementation (when tracing back errors). Although the authors suggested a few improvements (e.g. incorporating hard constraints) as well as improving its predictions in edge-cases (e.g. short-time horizons), by many measures this algorithm is state-of-the-art and it would be feasible to implement due to the level of documentation available with the Crocoddyl library.

The difficulty of implementing a closed loop controller is high since nonlinear systems are an unfamiliar topic at the graduate level. Since the dynamics of our robot model are unknown, this will be the second issue. The kinematics of our robot model is the only available information and this paper essentially depends on dynamics fully. The implementation of the Crocoddyl library into python will be fairly simple. The library is available with python bindings at the following link [crocoddyl · PyPI](https://pypi.org/project/crocoddyl/#data) Another option is the implementation of Pinocchio, which should be straightforward as well, since it can be downloaded from Github and can be implemented into Python. [Pinocchio (stack-of-tasks.github.io)](https://stack-of-tasks.github.io/pinocchio/)

***STOMP: Stochastic Trajectory Optimization for Motion Planning***

Mrinal Kalakrishnan, Sachin Chitta Evangelos Theodorou, Peter Pastor, and Stefan Schaal

**Intro**

The stochastic trajectory optimization framework utilizes noisy trajectories to explore costs of different paths, with obstacles taken into cost function, and uses these trajectories to update controls. The algorithm optimizes both smoothness (motor torques), and obstacles avoidance (calculated using SDF).

**Experiment**

In this paper, Stochastic Trajectory Optimization for Motion Planning (STOMP) utilizes optimization over a fixed duration T, discretized into N waypoints, equally spaced in time. Specifically, in order to optimize the motion planning of the robot manipulator, STOMP optimizes over a performance criterion rather than updating a gradient cost function, which is known to be susceptible to failure since it is a requirement for the cost function to be smooth and differentiable to be solvable. Rather than adding penalties, the checking for collisions involves a Series of Convex Optimizations (SCO), which solves approximations for the cost function and constraints of the problem. The possibility of adding constraints over costs allows for the probability of solving a larger subset of motion planning problems for nonholonomic systems. Recall, the nonholonomic systems have varying states depending on the trajectory that is taken. The algorithm utilizes the collision detection and numerical optimization scheme to account for optimized continuous time collision checking.

When implemented on a PR2 Mobile manipulation robot, much less collisions occur on the 7R robotic arm using STOMP for shelf-reaching tasks, due to the smoothness and optimized torque needed over time. This results in reaching lower minima, and not getting stuck within a local minima, which is the downfall of Differential Dynamic Programming (DDP) solutions. This paper has optimal results in optimization and obstacle avoidance.

**Evaluation**

One benefit of the STOMP algorithm is it works for non-differentiable / non-positive definite objective cost functions, which has proven optimal results. Furthermore, it has a simple to understand algorithm, as opposed to Model Predictive Control (MPC). Although, the algorithm’s computational complexity scales linearly with dimensions (need to take N dimensions).

The core libraries are implemented in C++, although python bindings are generated using Boost Python. There is open-source code available on Github, however as stated before, it is primarily implemented in C++. The team is not familiar with Sequential Convex Optimization (SCO) so the implementation of STOMP will be fairly difficult.

In comparison to high-frequency MPC, this algorithm appears less sophisticated, but stronger in different use cases (e.g. cost-function has many local minima, obstacles form convex space). The biggest benefit of this algorithm is the ease of the mathematical model behind the motion planning optimization implementation. Tradeoff, it is hard to tell which tasks this control model performs poorly on, as the author's evaluations emphasize areas where the model succeeds. After completing an evaluation of this paper, it is a very difficult path optimization algorithm to implement.

***Increasing Efficiency of Optimization based Path Planning***

Hao Ding, Gunther Reißig, and Olaf Stursberg

**Intro**

The plan of this paper is to propose a geometric result for path planning for a robotic manipulator interacting with obstacles, collisions, and kinematic and dynamic constraints are obeyed. The uniqueness of the paper is due to the time-varying obstacles and their positions over the prediction horizon for planning the path. This non-convex optimization problem can be approximated by Mixed Integer Programs (MIPs). In this paper, a geometric result whose application drastically reduces the number of binary decision variables in the aforementioned MIPs for 3D motion planning problems. This leads to a reduction in computational time, which is demonstrated for different scenarios discussed further below.

**Experiment**

This paper attempts an optimization implementation for a robot that interacts with 2 or 3 obstacles on a robot with a 2-link robotic manipulator. The dynamics of the robot within the paper are represented by bounds on the joint velocities for simplicity. They justify this by stating “ … joint positions of industrial robots are usually controlled by built-in lower-level controllers. Depending on the capability of the latter, bounds on velocities can be chosen to guarantee that the robot can successfully track the optimized trajectories.” The motion planning problem defines the workspace with the Euclidean Space for the non-convex optimization problem with Mixed Integer Programs (MIPs), a common topic in path planning and optimization. Although, the implementation of the motion planning problem can be difficult due to the dimension complexity in this paper. The motion planning algorithm is used to determine optimized trajectories of the joint positions in the workspace in order to get the end effector to its goal. The terminal state is reached utilizing non-convex optimization with MIP approximations. The paper discusses successful optimization over a short time horizon.

**Evaluation**

Mixed Integer Programs (MIP) is a new topic that will require the team to learn the underlying mathematics utilized in this algorithm. This paper is fairly difficult to understand when introducing the motion planning problem and the MIP theorem(s) together. The methods of implementation within the paper are as follows: implementation in C++ using Concert Technology libraries in CPLEX Studio Academic Research. The following has clear documentation from IBM. It is well documented using the following libraries: [Setting up the Python API of CPLEX - IBM Documentation](https://www.ibm.com/docs/en/icos/20.1.0?topic=cplex-setting-up-python-api), [Starting the CPLEX Python API - IBM Documentation](https://www.ibm.com/docs/en/icos/20.1.0?topic=tutorial-starting-cplex-python-api).

Although it is possible to use this library within python, from the evaluation, it seems difficult to carry out the method proposed in the paper and documented in the libraries above. The main concern is the time to get familiar with the topics within this paper, as the volume of documentation is quite large compared to the other libraries mentioned in this review, and begin to use it in this robot manipulator’s case.

**Conclusion**

**Key Factors**

When evaluating the research papers, the judging criteria lay in the following criteria: time constraints, complexity, efficiency of the optimization theory and results, and pitfalls/weaknesses the papers highlighted. The following allowed the team to judge the best optimization theory to adapt and implement for the time left in the internship, as well as complexity implemented as the work is at the undergraduate level. The evaluation describes the reasoning of the team based on the following judging criteria.

**Resolution**

In conclusion, the high frequency Model Predictive Control (MPC) is the paper selected for implementation. It seems to be the best option out of the 3 papers highlighted. Although to consider it to be the “best” option, it constitutes the least difficult to implement within the given constraints mentioned above. The team will need to learn the dynamics of the robot and apply closed loop control, since it is evaluated to be the best method within the results outlined in the paper. This closed loop control will rely heavily on sensor feedback. Trade off, considering the experimental results of the paper, the robot does not account for obstacles. However, one of the benefits of the paper is that the robotic manipulator is simulated in the KUKA iiwa, which is a similar model to the robot manipulator our team will be working with. In the team’s opinion, this paper was the easiest to understand, with optimal results, of the 3 papers researched and the best choice for the implementation of the Open Avenues Micro-Internship.