Progress meeting 9 | 7 October, 2022

1. Progress

- **Teleoperation:**

- **Custom end effector:**

- **Impact detection:**

- **Learning from demonstration:**

2. Agenda

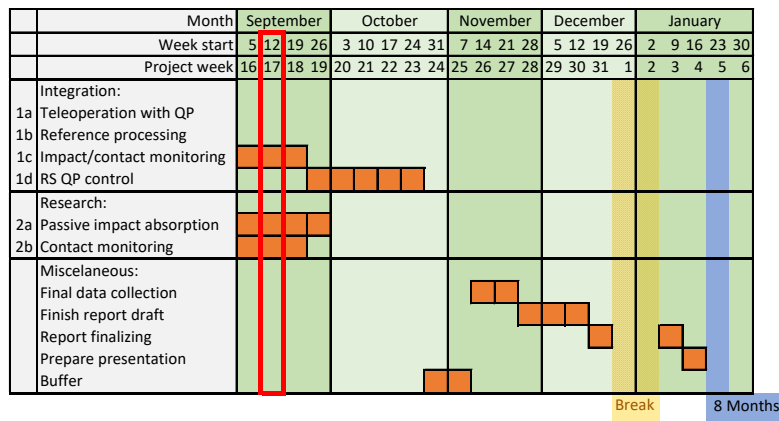During the meeting, I would like to discuss the following points:

-

-

3. Next steps

-

-

4. Long-term planning

Below is the current long-term planning for the project phase. The description of the labeled subjects are given in the preparation phase report and are also provided on the next page.

Month	September	October	November	December	January
Week start	5 12 19 26	3 10 17 24 31	7 14 21 28	5 12 19 26	2 9 16 23 30
Project week	16 17 18 19	20 21 22 23 24	25 26 27 28	29 30 31 1	2 3 4 5 6

Integration:
1a	Teleoperation with QP
1b	Reference processing
1c	Impact/contact monitoring
1d	RS QP control

Research:
2a	Passive impact absorption
2b	Contact monitoring

Miscelaneous:
Final data collection
Finish report draft
Report finalizing
Prepare presentation
Buffer

Break · 8 Months

1a **Translating dual-arm teleoperation to the physical setup.** The existing implementation in simulation already uses the mc_rtc interface, meaning that the switch to reality shouldn't pose an issue. Nevertheless, this step also involves getting familiar with the software, which increases the anticipated time for this step.

1b **Extracting references from the demonstration data.** The demonstrated trajectories should be split into ante-impact and post-impact sections, and extended to facilitate RS. Furthermore multiple measurements should be used to fit ProMPs, after which

1

a reference can be generated. It is also key to identify which data should be learned from the demonstration. This is not limited to choosing between a force or position reference, but can also consists of learning properties of the environment, e.g. friction cones or box inertia, that are crucial to a dual-arm box grabbing scenario.

1c **Integrating impact detection and contact monitoring in mc_rtc.** The majority of the impact detector's complexity resides in the momentum observer; however, Franka Emika's software already has an integrated momentum observer. This still leaves tuning of the impact detecting algorithms which might be time consuming. Furthermore an analysis comparing the available methods could be worthwhile. Factors which determines the effectivity of the impact detection algorithm include speed of detection, as well as reliability, i.e. the rate of false positives. The addition of objects that cause unexpected impacts is not considered a part of the research scope.

1d **Configuring QP controllers for the ante-impact, intermediate, and post-impact phase.** For each of the phases, it is important to address the redundancy in the arms' degrees of freedom. After that, control for the ante-impact phase should be trivial. For the intermediate phase, it is expected that ante-impact reference tracking without velocity feedback should be applicable on a dual-arm robot, though this might prove to be false, in which case other methods should be investigated. During post-impact control, the challenge will be maintaining non-slip contact with the box. It is difficult to say how well the results from simulations can be repeated with torque control, where the state of the box can not be sensed to be used in the QP controller.

2a **Passive impact absorption:** A soft cover for the end effector will be designed. Such a cover can be connected to the Panda by connecting bolts to the so-called flange interface. A mold will be created using 3D printing to allow for casting of various silicone soft covers. Design parameters – i.e. material properties (controlled by choosing different kinds of silicone) and soft cover thickness – will be analyzed experimentally. A systematic comparison between various designs will require an experiment plan including a realistic testing scenario. Evaluation of performance can be based on the oscillatory response in position and force after establishing contact. Furthermore multiple scenarios with various box surface properties and robot poses should be considered.

2b **Contact monitoring:** When investigating contact monitoring, two approaches can be taken: either using proprioceptive or exteroceptive sensors. A possible improvement for contact monitoring using proprioceptive sensors could be to wait a fixed time starting from the last detected impact, rather than waiting a fixed time from the first impact. As for exteroceptive sensors, they can be a hurdle for large-scale commercial applications as they are not integrated in the robot. However, if a soft cover is to be mounted to the end effector, including tactile sensors for impact and contact monitoring becomes more feasible. Practical questions such as which tactile sensor to use and how to integrate it could be addressed, though this is not absolutely necessary for completing the research goals, and therefore has a low priority.

1 Impact detection using contact force vs velocity

For impact detection, Benn and Sven considered two options: looking at jumps in velocity, or jumps in force. Ultimately, Benn and Sven both ended up using the forces, whereas I think looking at velocities makes more sense in the context of reference spreading. This section explains my reasoning.

Looking at jumps in force for impact detection could result in false positives. Consider an end effector that is already exerting a contact force on a fixed surface using impedance control. A jump in the reference would result in a jump in the contact force. Even if contact is maintained throughout the event, the jump in contact force would trigger the impact monitor while no impact has actually occurred. In the described situation, the velocity remains unchanged, meaning that a velocity-based impact detector would not return a false positive.

Additionally, contact forces cannot be directly measured, and the process of estimating them with a momentum observer is quite complex and involves a delay when compared to determining the velocity.

Given the downsides of looking at force rather than velocity, there must be significant benefits to warrant using force estimates for impact detection. I am not able to fully identify Benn's reasoning here (he states that "The most common method of impact detection is through the use of external joint torque estimators", though I suspect this is in the context of human safety). Sven, however, does provide an argument. In his experiments, he noticed that velocity data was missing during some timesteps, which resulted in poor impact detection. The impact detector using force did not suffer from this issue.

As discussed during the questions of Sven's defense, the momentum observer acted as a filter which made it so that missing data was not problematic for the impact detector that uses force. Therefore I don't think that missing velocity data is a valid argument for looking at force data; applying a filter to account for missing velocity data makes more sense in my opinion.

As a final argument, the reason for applying reference spreading is diminishing peaks in the velocity tracking error. If a jump in velocity cannot be measured, then it is also not possible for the velocity tracking error to peak (barring a peak in the reference). While there is some connection between the peaking error and contact forces, the connection between measured velocity and the peaking error is much closer.

Conclusively, based on the arguments given above, a velocity-based impact detector is more appropriate for reference spreading than a force-based impact detector.

2 Learning impedance vs force/position reference

When considering what reference to extract from demonstrations, two options come to mind. Firstly, it is possible to look at the position trajectory of the robot during demonstration. As the robot position by itself does not fully describe the robot's state, a force reference should also be extracted. This resembles the approach used by Sven. A second option would be to learn an impedance reference, i.e., the position of the VR device, which encapsulates both position and force. This section describes how both approaches would be implemented for a simple testcase. Furthermore, the pros and cons of each method will be discussed.

As a disclaimer, it should be noted that neither Sven's implementation nor the current VR implementation utilize a velocity reference. While Sven used reference spreading for the force reference, the value of reference spreading might be more apparent when applied to a velocity reference. The tracking performance is also expected to improve with the addition of a velocity reference. Therefore, the remainder of this section will assume that a velocity reference is included in the control scheme. Obtaining this reference should not prove difficult as a robot's velocity can easily be determined, and the HTC Vive provides velocity (and even acceleration) measurements.

2.1 Testcase description

The testcase consists of a 1-dof system with mass $m = 1$ kg and position x. The EoM is given by $m\ddot{x} = F_c + F_e$, with control force F_c and environmental contact force F_e following

$$F_e = \begin{cases} 0 & \text{if } x \geq 0 \\ -k_e x - d_e \dot{x} & \text{if } x < 0. \end{cases}$$

Parameters $k_e = 10^4$ N/m and $d_e = 10$ Ns/m represent the environment's contact stiffness and damping, respectively.

The control force during the demonstration, F_c^d, is generated using an impedance controller with stiffness $k_c = 40$ N/m and damping $d_c = 2\sqrt{k_c}$ Ns/m:

$$F_c^d = k_c(u - x) + d_c(\dot{u} - \dot{x}).$$

The references u and $\dot{u}$ emulate an input coming from the VR device.

Figure **??** shows the reference u, together with simulation results of the demonstration. Five signals from this simulation are saved, so that they may be used as a reference for the LfD controllers: u, $\dot{u}$, x^d, $\dot{x}^d$, and F_e^d. Superscript d denotes a result from the demonstration simulation.

It should be noted that determining F_e^d for use in a reference is not easy outside simulations. At first sight, it might look like F_e^d can be found using the EoM following $F_e = m\ddot{x} - F_c$, however, $\ddot{x}$ is not measured directly. A momentum observer could be used, though this results in a first-order filtered estimation. It is speculated that this estimation can be improved using Kalman smoothing. In an experiment of length T, a Kalman filter estimates $F_e^d(t)$ (with $t < T$) using data collected in interval $(0, t)$. Kalman smoothing runs offline, and therefore has the benefit of using data from the entire experiment, i.e., interval $(0, T)$. This could drastically improve estimations.

2.2 LfD controllers without reference spreading

Firstly, the controllers are described without reference spreading. The impedance controller F_c^i is identical to the controller used in demonstration, i.e.,

$$F_c^i = k_c(u - x) + d_c(\dot{u} - \dot{x}).$$

Furthermore, the force/position control force F_c^f is given by

$$F_c^f = k_c(x^d - x) + d_c(\dot{x}^d - \dot{x}) + F_e^d.$$

This force/position controller is identical to what Sven used, with the exception that Sven did not use a velocity reference $\dot{x}^d$. (He does fit a reference to the velocity, but it is not used by his controller according to his paper/thesis. Could this perhaps be an oversight?)

These controllers were evaluated in an ideal scenario, meaning that the impact occurs at the exact same time as it did during the demonstration, and there is a perfect estimation of F_e^d. The results are shown in Figure **??**.

For the impedance controller, the results exactly match the demonstration as there is no difference between the controllers.

The position/force controller shows drastically different behavior however. Already in the ante-impact phase where the force reference is still zero, the controller tracks x^d which tracked u, meaning that x^p will always lag behind. This could be improved by directly tracking u in the ante-impact phase.

Furthermore, at the moment of impact, the force controller tries to track the peak in the force trajectory while, at the same time, the impact itself also causes a peak in force. As a result, F_e^p spikes to approximately $2F_e^i$ at impact. After impact, contact loss due to bouncing is far more evident in x^p than it is in x^i. Despite these results, the force/position controller should not yet be disregarded without looking at a situation where reference spreading is required.

In a less ideal scenario, estimation $\tilde{F}_e^d$ is a first-order filtered version of F_e^d following

$$\dot{\tilde{F}}_e^d = k_f(F_e^d - \tilde{F}_e^d)$$

with gain $k_f = 10 \text{ s}^{-1}$, and $\tilde{F}_e^d(0) = F_e^d(0)$. The filtered signal is shown in Figure **??**. This change does not affect the impedance controller, however the force/position controller shows significantly improved results in Figure **??**, since the filter removes peaks in the force reference.

2.3 LfD controllers with reference spreading