Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

SOFTWARE DEVELOPMENT PROJECT (SDP)

TOPIC: MOTION CONTROL OF THE KELO 500

Team:

Sivva Rahul Sai

Kavya Shankar

Kishan Ravindra Sawant

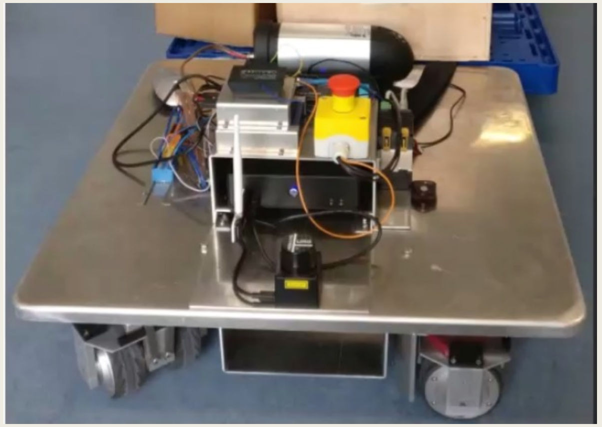
Supervisor:

Sven Schneider

March 21, 2022

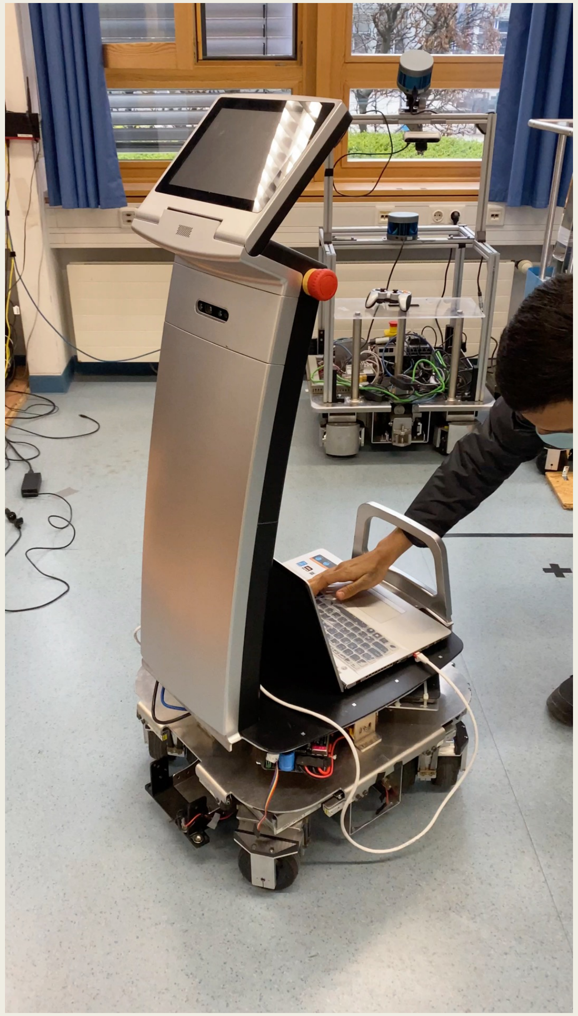
What is KELO 500?

- KELO 500 is an **over-actuated** mobile platform
- Four identical pair of wheels, which can be actuated independently
- Passive **back drivability** is a feature of this design
- Wheel actuation depends on **desired motion** and other **policies**
- Communication with wheel-units (slaves) is made over **EtherCAT**



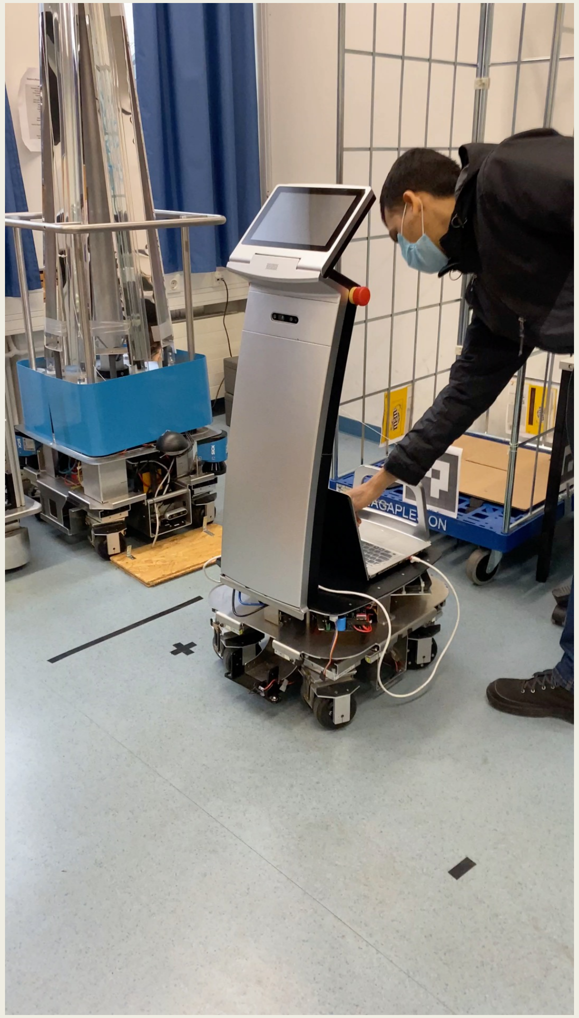
KELO 500 robot

Motivation



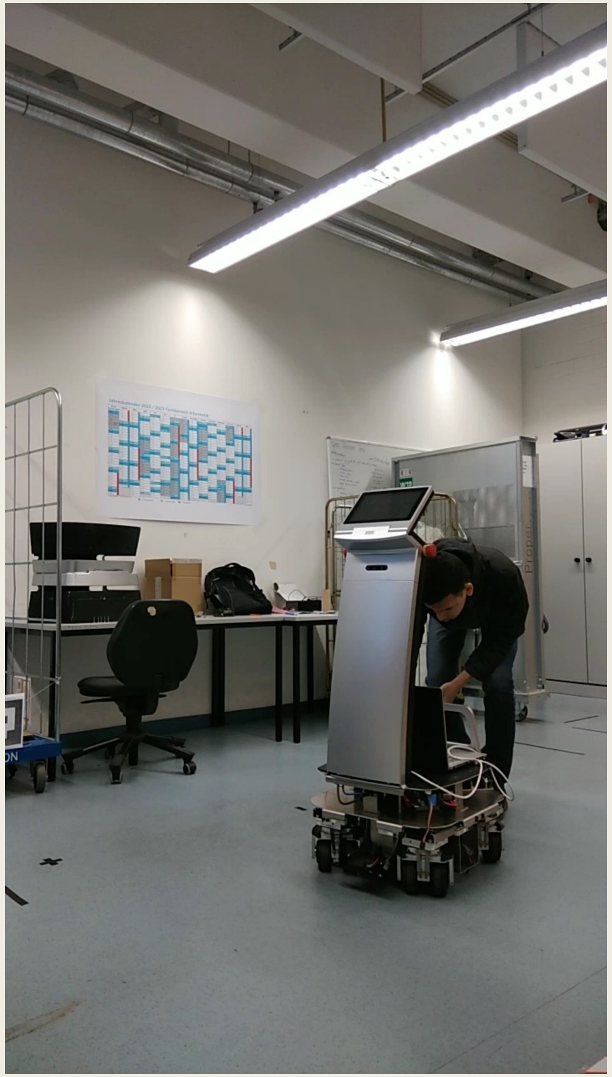
Rotation about the axis

Straight motion in +y direction



Diagonal motion

How is it different from velocity control?



Demonstration of motion in torque control mode

Project Goals

- **Approach**
  - Resolution of redundancy in the over-actuated platform
  - Design control policies to improve efficiency of force transmission
  - Using suitable numerical-solver to calculate force distribution

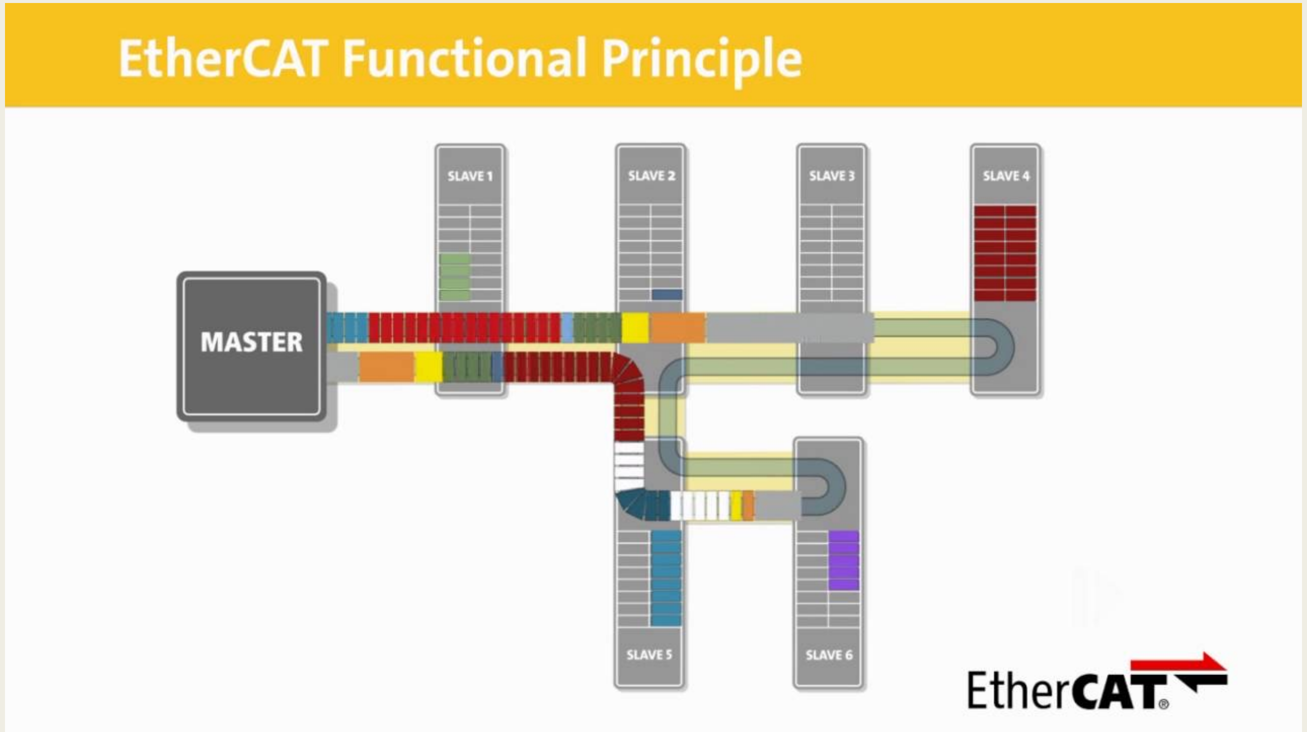
- **Implementation & Integration**
  - Integration of BLAS/ LAPACK and SOEM libraries
  - Implementation of numerical force distribution solver with GSL
  - Implementation of EtherCat/communication interface with SOEM
  - Integration of solver and communication interface into application
  - Integration of control policies with the software

User stories

Unique Identifier	User Stories	Acceptance Criteria	Priority	Estimate	Risk	Real Effort
US1	As a Developer I want to establish EtherCAT connection via SOEM library So that I can query information about the clients (wheels)	Find out IDs of all the clients (wheels) connected via EtherCAT	Medium	1w	Low	1w
US2	As a Developer I want to send control commands to the wheels So that I can identify their association between the slave IDs and their geometric location on the platform's base	When a wheel is given a control command, it should follow the command	Medium	2w	High	2w
US3	As a Developer I want to provide same torque/ velocity to all the wheels So that I can test the hypothesis that all wheels align in the direction of motion	Wheels do align in the direction of motion (accept hypothesis) Wheels do not align in the direction of motion (reject hypothesis)	Medium	1w	Low	2w
US4	As a Developer I want to • Find the forward force matrix • The mapping of wheel torques to the force generated at individual revolute joints So that I can calculate the platform force generated by a combination of torques at individual wheels	The values derived from manually calculations should match with that of the software	High	1w	High	2w
US5	As a Developer I want to determine the pseudo inverse of the forward force matrix (G) using LAPACK So that I can map Cartesian platform force to the individual wheel torques	The values derived from manually calculations should match with that of the software	High	2w	High	3w
US6	As a Developer I want to compute the 'weighted pseudo-inverse' of the force mapping by providing the weights So that I can minimize the internal forces to achieve efficient motion	Providing zero weights to individual wheels should make them passive Varying weights should vary the torque proportionately	High	3w	High	4w

SOEM Library

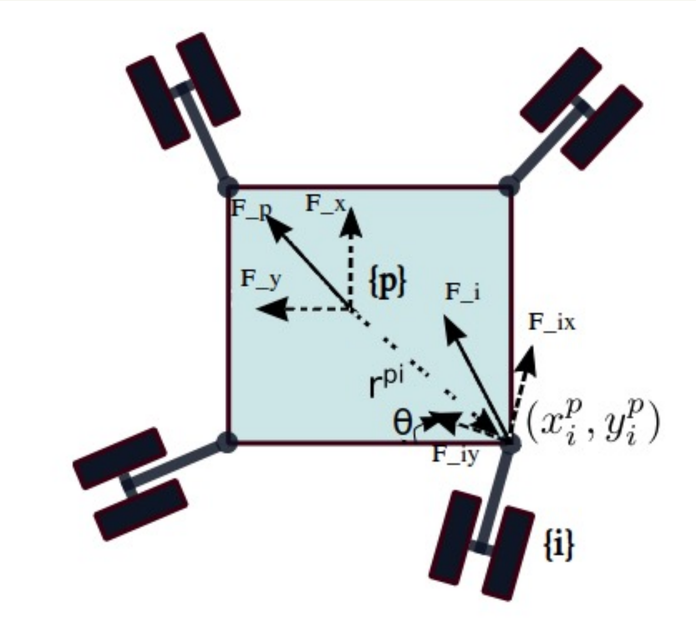
- SOEM (Simple Open EtherCAT Master) is an open source library used to perform EtherCAT communication.
- SOEM library is used to transfer control commands to the wheels units and receive pivot information in response.



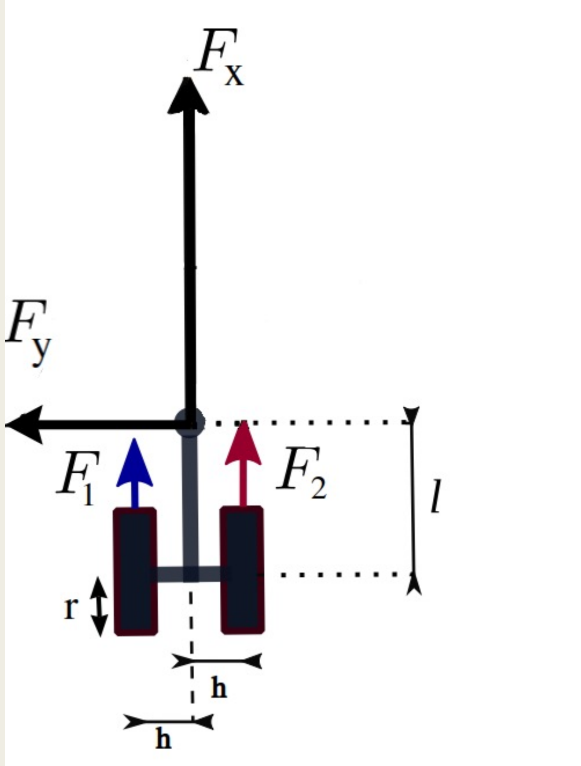
EtherCAT communication [2]

Physical Interpretation

- Platform forces F^p is mapped to wheel torques
- Individual forces at the corners are determined by inverse force kinematics
- Pivot forces are mapped to individual wheel forces
- Wheel forces are mapped to torques



4 wheel-units platform [1]



Pivot-wheel force distribution [1]

Mathematical Interpretation

$$\underbrace{F^p}_{3\times1} = \begin{pmatrix} F_x^p \\ F_y^p \\ M^p \end{pmatrix} = \underbrace{G}_{3\times8} \underbrace{\begin{pmatrix} F^1 \\ \vdots \\ F^4 \end{pmatrix}}_{4\times(2\times1)} = G \begin{pmatrix} F_x^1 \\ F_y^1 \\ F_x^2 \\ F_y^2 \\ F_x^3 \\ F_y^3 \\ F_x^4 \\ F_y^4 \end{pmatrix}$$

Relation between platform force (F^p) and corner forces (F^i) [1]

$$G = \begin{bmatrix} c_1 & -s_1 & c_2 & .. & -s_4 \\ s_1 & c_1 & s_2 & .. & c_4 \\ x_1 s_1 - y_1 c_1 & x_1 c_1 + y_1 s_1 & x_2 s_2 - y_2 c_2 & .. & x_4 c_4 + y_4 s_4 \end{bmatrix}$$

Forward force matrix (Jacobian) [1]

Inverse force kinematics

$$B = \underbrace{W}_{3X3} * \underbrace{G}_{3X8} * \underbrace{K}_{8X8}$$

$$B = U * \Sigma * V^T$$

$$G^+ = K * V * \Sigma^+ * U^T * W$$

$$\underbrace{G^+}_{8X3} * \underbrace{F^p}_{3X1} = \underbrace{\begin{bmatrix} F^1 \\ . \\ . \\ F^4 \end{bmatrix}}_{8X1}$$

Pivot force calculation

$$\tau_1 = 0.5r\left(F_x - F_y\left(\frac{l}{h}\right)\right)$$

$$\tau_2 = 0.5r\left(F_x + F_y\left(\frac{l}{h}\right)\right)$$

Torque calculation

GSL - GNU Scientific Library

- GSL library is an open source library.
- Used for numerical computations in applied mathematics and science
- Provides support for BLAS and is well documented
- The GSL is written in C
- For linear algebra we have used gsl matrix and gsl vector datatypes

Lessons learnt

- Verification of conventions used in reference model with the actual model
- Consideration of time requirement for integration of the algorithms and debugging
- Verifying different functionality of the robot after integrated is more spontaneous
- Physical systems have small deviations from expected ideal behaviour
- Use of python for prototyping mathematical models saves considerable time and effort
- Documentation of operating robot and version control of code is beneficial for debugging
- Verifying controllable and uncontrollable safety features

Reference for future developers

- Understanding and establishing EtherCAT connection requires prior knowledge or supervision.
- While developing mathematical model, consistent verification by backtracking is beneficial
- Use of python for prototyping mathematical model
- Considering safety measures before running the robot
- Documentation of running the robot and tracking code updates
- Standard coding practices and naming conventions helps for better maintenance

Communication and tools

- Daily 15 minutes scrum meeting focus (development team)
  - *What did we do yesterday?*
  - *What will we do today?*
  - *Is there any impediment?*

- Weekly meeting with supervisors (1-2 hours)

- Tools used for collaboration
  - *Webex (weekly meetings)*
  - *Skype (daily team meetings)*
  - *Github (Version control)*

Future work

- Adapt the implementation to suit for other robots (eg: RObile)
- Adapt to setup where some of wheel units are passive
- Support for joystick control
- Implementing using ROS/motion planner
- Dynamically update the weights when a wheel loses the connection
- Addition of configuration files

Reference

[1] Bruyninckx, H. (2021). Design of Complicated Systems. Work in progress. URL: https://robmosys.pages.gitlab.kuleuven.be

[2] https://www.youtube.com/watch?v=z2OagcHG-UU

THANK YOU