OULUN
YLIOPISTO

TEKNILLINEN TIEDEKUNTA

MULTI-ROBOT COOPERATION

Markus Heikkinen

KONETEKNIIKAN TUTKINTO-OHJELMA

Kandidaatintyö 2019

TIIVISTELMÄ

Usean robotin yhteiskäyttö

Markus Heikkinen

Oulun yliopisto, Konetekniikan tutkinto-ohjelma

Kandidaatintyö 2019, 32 s.

Työn ohjaaja(t) yliopistolla: Yrjö Louhisalmi

Tässä opinnäytetyössä perehdytään usean robotin yhteiskäyttöön, jossa mielenkiinnon kohteena on kahden robottikäden yhteistoiminta. Työ on kirjallisuuskatsaus. Robottien toimintaa ja niihin vaikuttavia asioita tarkastellaan niin kinematiikan, kuin ohjelmisto- ja käskyarkkitehtuurin kautta.

Työn pohjana käytettiin yliopistolla olevien KUKA robottikäsien oppaita, jotta jatkossa niiden yhteiskäyttö olisi helpommin ymmärrettävissä.

Työn tulokset avasivat sitä, miten robottien ohjelmisto ohjaa ja laskee tarvittavat liikeradat ja geometriat ja mitä kaikkea usean robotin ohjauksessa pitää ottaa huomioon.

Tämä on hyvä pohja syvemmälle teoreettiselle robottiohjelmistolle tai käytännön testaamiselle.

Asiasanat: robotti, KUKA, yhteiskäyttö, kinematiikka, esineiden internet

ABSTRACT

Multi-robot cooperation

Markus Heikkinen

University of Oulu, Degree Programme of Mechanical Engineering

Bachelor's thesis 2019, 32 p.

Supervisor(s) at the university: Yrjö Louhisalmi

This bachelor's thesis familiarizes with multi-robot cooperation. The main interest is in two robot manipulators. This thesis is a literature review. The operation of the robot and the phenomena that act on them while in operation are investigated from kinematics and command architecture point of view.

This thesis is based on manuals from two KUKA robots from University of Oulu, so in the future the use and understanding of their cooperation would be easier.

The results gave good understanding of robot software calculations for trajectories and geometrics and what other has to consider when controlling a multi-robot system.

This is a good base for deeper theoretical research for robot system software and practical testing.

Keywords: robot, KUKA, cooperation, kinematics, networked robotics, internet of things

PREFACE

Industrial robots are becoming more and more common. Today's manufacturing methods need complex multi robot systems. The goal was to learn more about the system and what the robotic software does during operation.

The first idea for the thesis was to make practical tests with the school's robot arms, but due time restrictions the topic was changed to literacy review. This thesis was written during spring 2019.

I want to thank my supervisor Yrjö Louhisalmi for guidance, since robots and their cooperation is a broad subject and it was hard to narrow to specific problems.

Oulu, 07.05.2019

Markus Heikkinen

Table of Contents

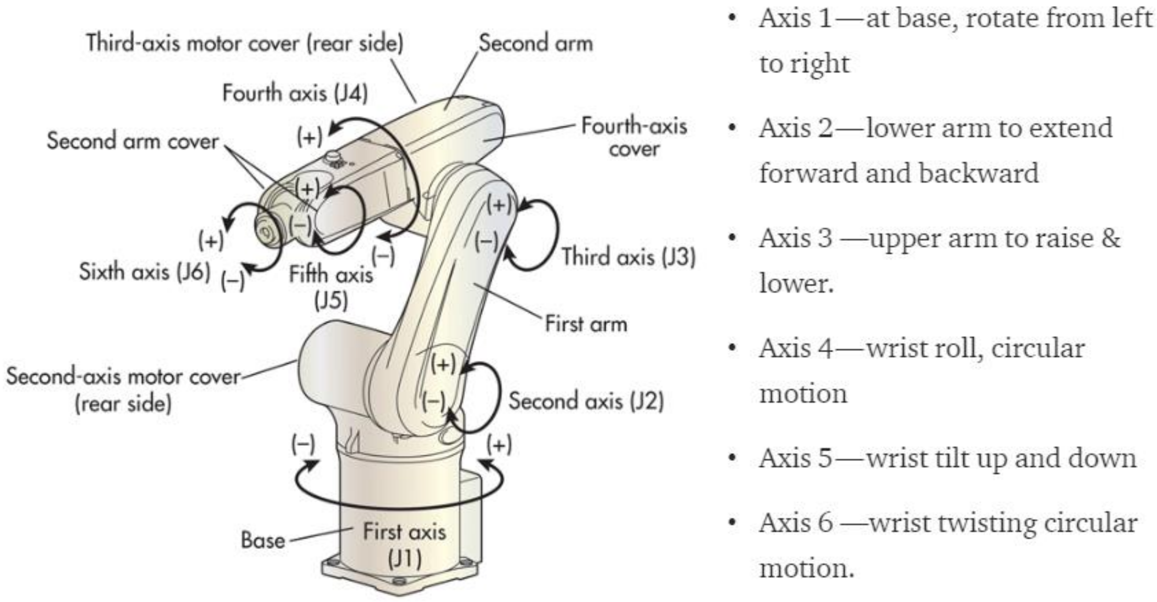
SYMBOLS AND ABBREVIATIONS

A	Agent
CAD	Computer Aided Design
CPSs	Cyber Physical Systems
CPU	Central Processing Unit
DOF	Degrees Of Freedom
E	Environment
IoT	Internet of Things
MAS	Multi-Agent System
MRS	Multi-Robot System
O	Object
OEM	Original Equipment Manufacturer
O_p	Operation
OS	Operating System
P	Reference point P
R	Relations
R_i	Robot i
RTOS	Real-Time Operating System
SoA	Service-oriented Architecture
TCP	Tool Center Point

1 Robot arm movement

1.1 Axes

6-axis robot means that the manipulator has six degrees of freedom (6 DOF): forward/back, up/down, left/right, yawn, pitch and roll. The possible movements of 6-axis robot arm are shown in figure 1.



- Axis 1—at base, rotate from left to right
- Axis 2—lower arm to extend forward and backward
- Axis 3 —upper arm to raise & lower.
- Axis 4—wrist roll, circular motion
- Axis 5—wrist tilt up and down
- Axis 6 —wrist twisting circular motion.

Figure 1. 6 DOF robot arm. (David J Bland 2017)

1.2 Kinematics

In multi-robot cooperation, especially manipulators, finding the right control system for the work space at any given task is fundamentally challenging problem. This requires exact knowledge of the physical nature of the system and its mathematical basis. Kinematic and force uncertainty in mathematical description are crucial problems. It might be impossible to uniquely determine the exact parameters for contact forces, driving torques and kinematic quantities. The problems listed before are a common topic in robotic research. (Živanović, Vukobratović 2006, preface)

Movement of a 6-axis robot arm is complex, and technical details are not included in this thesis, but it's worth noting that understanding of different kinds of kinematics, dynamics and joint mechanisms is essential to when designing the robot manipulators or multi-robot systems. *Since robotic mechanisms are by their very essence designed for motion, kinematics is the most fundamental aspect of robot design, analysis, control, and simulation* (Oussama Khatib 2008, p. 9).

1.3 Forward kinematics

When the operator wants to setup the end effectors coordinates of the robot arm, it can be done by moving the reference frame of end effector compared to the base of the robot's reference frame. The joint positions relative to the base are then calculated with forward kinematics. *This is straightforward for a serial chain since the transformation describing the position of the end-effector relative to the base is obtained by simply concatenating transformations between frames fixed in adjacent links of the chain* (Oussama Khatib 2008, p. 26).

1.4 Inverse kinematics

The inverse kinematic problem is for a serial chain manipulators, like 6 DOF robot arms, to find the positions and orientations of the joints relative to end-effector and the base. Oussama Khatib (2008, p. 27) describes inverse kinematics with the following sentence: *"A more general statement is: given the relative positions and orientation of two members of a mechanism, find the values of all of the joint positions."*.

Most industrial robots are deployed with closed control architecture, that allows only kinematic control. It means, that the user can only modify outer velocity and position references to the low-level joint controllers. Usually there are no more information about the robot's dynamic model, structure or control parameters. Robot programming is done with manufacturer's communication interface. This is made so the end-user does not need to know the exact parameters to control and program the robot. (Geravand, Flacco et al. May 2013)

1.5 Joint space control scheme

Joint space control controls joint actuator forces. If the robot is operated with electric motors with reduction gear of high ratios, it reduces the nonlinearity of the system. The downsizes are joint friction, elasticity and backlash, that might limit the performance due inertia and centrifugal forces. (Sciavicco, Siciliano 1996, p. 213)

Direct drives can eliminate the drawbacks above, but weight of nonlinearities and couplings between the joints becomes relevant. This encourages the use of different controls in different systems. Figure 2 shows the close loop control loop for the joint space control. (Sciavicco, Siciliano 1996, p. 214)

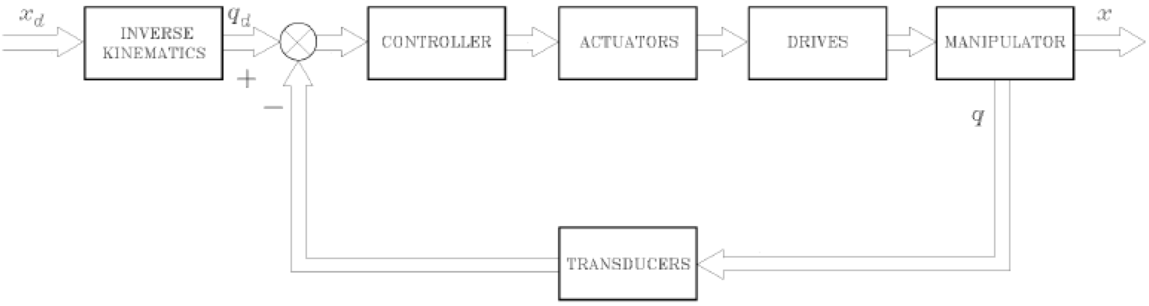


Figure 2. General joint space control scheme. (Sciavicco, Siciliano 1996, p. 214)

1.6 Operational space control scheme

End-effector motion and forces are present in operational space. This requires more complex algorithms than joint space control. Figure 3 shows that the inverse kinematics are embedded into the closed loop system. This is an advantage since the kinematics are usually calculated in the joint space controls. (Sciavicco, Siciliano 1996, p. 214-215)

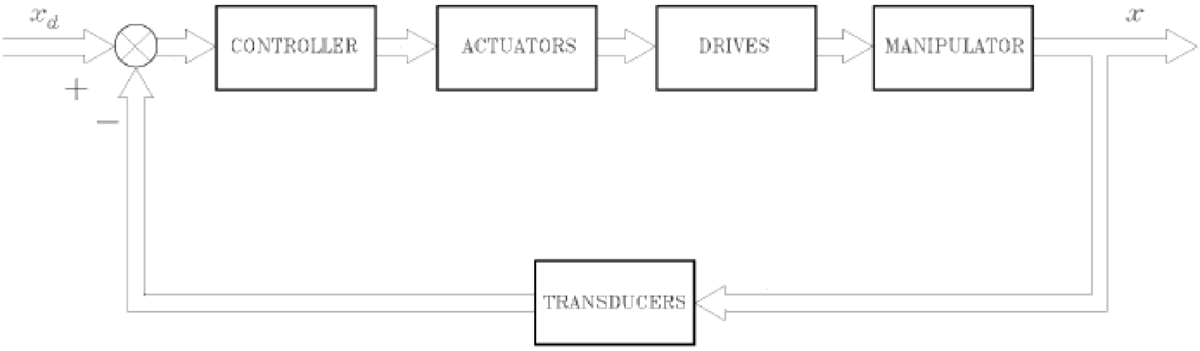


Figure 3. Operational control scheme. (Sciavicco, Siciliano 1996, p. 215)

1.7 Workspace

Most generally, the workspace of a robotic manipulator is the total volume swept out by the end-effector as the manipulator executes all possible motions (see Picture 4). The workspace is determined by the geometry of the manipulator and the limits of the joint motions and it defines the positions and orientations that it can achieve to accomplish a task.

The workspace can be classified as dexterous or reachable workspace. Reachable workspace is defined by the set points in a space that the reference point P in the end-effectors wrist can reach. Dexterous workspace is defined by the set points the reference point P with arbitrary end-effector can reach.

The wrist joints determine the orientation range (dexterous workspace) and they can travel from 360° up to 720°. In theory the revolute joints can have unlimited rotations, but there must be limits due to physical constraints, like wire entanglement. The task space of the robot can be increased by adding specialized tools, like grippers or calibration tools, to its end-effector.

Six degrees of freedom are the minimum required to place the end-effector or tool of a robotic manipulator at any arbitrary location within its accessible workspace. (Oussama Khatib 2008, p. 25, 68-69, 78)

With an increasing adaptation of flexible manufacturing systems and the need to reduce setup and launch times, it is important to know beforehand the possible limitations of a robotic manipulator, eliminating the need for trial and error and repeated adjustments in either the virtual or physical domains (Gudla 2012).

1.8 Work envelope

Work envelope (reachable workspace) is the volume of space the robot occupies while it is moving (Figure 4). The envelope is defined by the types of joints, their range of movement and the lengths of the links that connect them. Designing the paths and tasks for the robot, it's work envelope restrictions must be considered. The possible restrictions can be structural, acting loads, joint travel range, link lengths and angles between the axes. (Oussama Khatib 2008, p. 68)

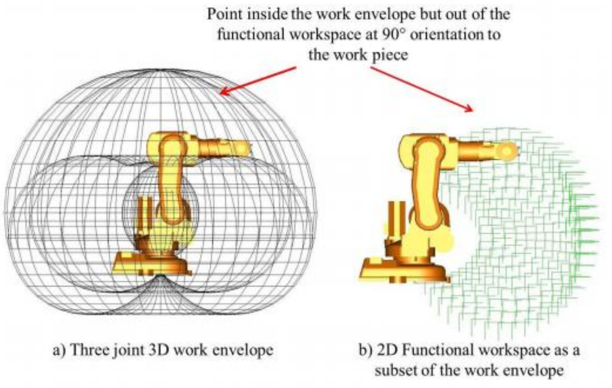


Figure 4. Limited 6-axis robot workspace and work envelope. (Gudla 2012, p. 3)

2 Multi-robot systems

In today's industry the use of multiple manipulators in a shared workspace is common and it creates many different problems to be taken care of. But it also extends the capability of the tasks to be made compared to invidual robots. Multi-robot teams can solve tasks that are impossible for invidual robot and they can be more reliable, cheaper and faster. In certain applications adding more robots into the system is superaddive, the whole is bigger than the sum of parts (Oussama Khatib 2008, p. 927).

Advanced industrial automation requires robots to be versatile and their control systems to be easily reprogrammable. They need to be able to make different batches with minimal downtime. (Sciavicco, Siciliano 1996)

Fully automatic robot cells move objects from robot arm to robot arm or they move an object together in load sharing applications in an overlapping workspace and the algorithms controlling the robots must be robust to avoid accidents and downtimes.

Optimal placement of robots in robot cells is important feature and if it is solved it can results in substantial cost and time savings. (Gudla 2012, p. 9)

The optimal and collision-free coordination of multiple manipulator robots in a shared workspace while considering their dynamics is an important open problem. There are several applications that involve this coordination task. Consider scheduling the motions of multiple robots in a welding or assembly work cell to minimize the cycle time. (Akella, Jufeng Peng 2004)

Multi-robot systems (MRS) cannot be generalized by single robot case and every approach or system must be precisely characterized about the environment and internal system organization, for example, multi-robot team's communication architecture. MRS can't also be considered as a special case of multi-agent system (MAS), because of uncertainties in the environment, sensor information and the quality of acquired data. (Farinelli, Iocchi et al. 2004)

2.1 Goals and tasks

The term 'multi-robot systems' can be used to refer to a wide range of robotics systems incorporating more than one robot, including swarms of many robot systems. Robot system can be assigned to perform a task. The task can then be broken into separate subtasks.

The main objective of robot cooperation is to manipulate an object. This contains the following goals:

- Changing the space position of an object, like transferring it.
- Tracking the given trajectory of the object at given orientation along the trajectory.
- Performing some work on a stationary or mobile object.

2.2 Homogeneous teams

In homogeneous teams the robots can perform the same functions within the system. Every task and its subtasks can be allocated to any robot in the system. Homogeneous teams allow flexibility and security in the system, if one robot fails, the others can perform its task. There are certain behavior-based strategies to monitor sensors of robots and performance and other robots can act accordingly if a fault is found. There must be fault tolerances in place for robot systems. The main issue is the distribution of the robots between the different tasks.

As opposite to homogeneous teams, heterogeneous teams of invidual robots are different and can perform only certain tasks and functions. Some robots can only move certain weight, or their workspace and position capabilities are different than others. The issue is to match the right task for the right robot. (Toshiyuki, Kazuhiro 2011)

2.3 Robotic systems architecture

Robot systems software is complex, especially for multi-robot systems. The sensors and actuators outputs and inputs, errors, noises and many other parameters must be controlled and calculated. A robot system needs an architecture to control all the variables, tasks and situations. Currently there is no single architecture that is suitable for all the applications. Different architectures have different advantages and disadvantages. The application usually determines what is the best system architecture for it and how it should be implemented. Robot architecture planning should not be taken lightly. The goal of the architecture is to make programming of a robot easier, faster, safer and more flexible. Implementing new architectures in old systems or trying to replace faulty one can be tedious and time consuming. (Oussama Khatib 2008, p. 1371)

2.3.1 ALLIANCE software architecture

When multiple autonomous robots are involved in a same workspace, task or environment, there must be command architecture in place. L.E. Parker introduces ALLIANCE software architecture. ALLIANCE is behavior-based architecture that is based on mathematical models to allow robots to act accordingly in a dynamic environment. The architecture is based on mobile robot teams, who need robust mapping, communication and actions based on their location, ability to complete assigned tasks and maximize the fault tolerance. There is no centralized control for the teams. The distributed robotic system must accomplish its goal by cooperating with other autonomous robots. Behavior based systems are flexible, reliable and easily modified and ALLIANCE adds fault detection and reactions to it. (L.E. Parker 1998)

ALLIANCE tries to mimic human behavior in teamworking. Usually many humans with different skills and specializations are more productive and effective than one human. The architecture emphasizes the importance of using different robots working on specific tasks in a team, but also having the ability to help and react to other problems of team members. ALLIANCE can also be used in a manufacturing cells that contains multiple manipulators working together.

L.E. Parker (L.E. Parker 1998) introduces 8 assumptions that were made while developing the ALLIANCE architecture. The assumptions are for small- to medium-sized teams of heterogeneous robots performing missions composed of independent subtasks that may have ordering dependencies.

1. The robots on the team can detect their own actions, with some probability greater than 0.

2. Robot r_i can detect the actions of other team members for which r_i has redundant capabilities, with some probability greater than 0; these actions may be detected through any available means, including explicit broadcast communication.

3. Robots on the team do not lie and are not intentionally adversarial.

4. The communications medium is not guaranteed to be available.

5. The robots do not possess perfect sensors and effectors.

6. Any of the robot subsystems can fail, with some probability greater than 0.

7. If a robot fails, it cannot necessarily communicate its failure to its teammates.

8. A centralized store of complete world knowledge is not available.

ALLIANCE breaks down mission to tasks and subtasks. Robots itself evaluate their ability to perform certain tasks based on their skill levels. The robots in the system receive continuous feedback and data from sensors and actuators. Actuators work based on the motivation of the robot (Figure 5). The agents in the system have two internal motivations: impatience and acquiescence. Impatience motivation enables robot to handle situations other robots fail to perform and acquiescence enables robot to allocate tasks it itself fails to perform. (L.E. Parker 1998)

While the ALLIANCE architecture is developed for mobile robots, the architectural structure and behavior control can still be implemented to systems like robot arms operating in manufacturing.

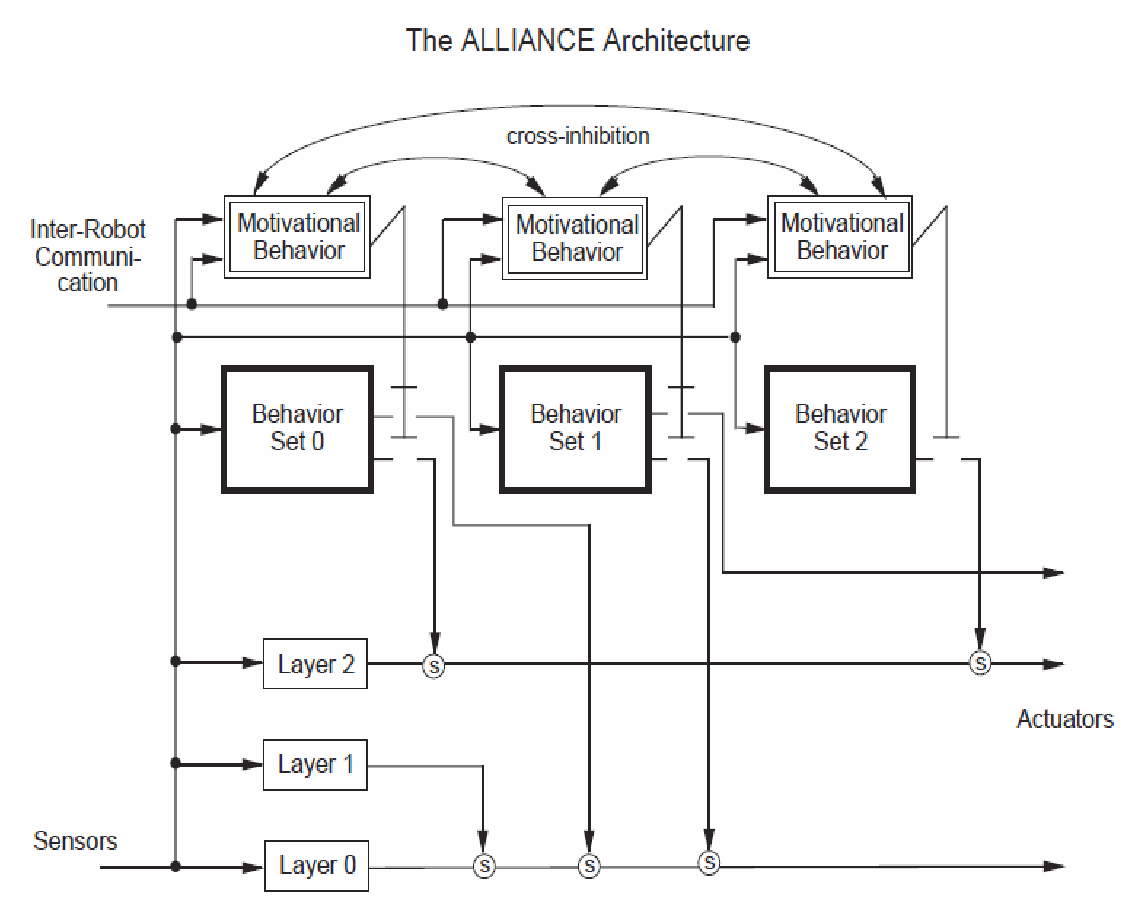


Figure 5. The ALLIANCE Architecture. (L.E. Parker 1998)

3 Networked robotics

We can use the term networked robotics as defining the field treating robotic systems where some information is exchanged among system components through a network. The challenge is to find only the essential information and data to be send through the network, because there is nonlinear relationship between the information communicated and its impact on performance. (Oussama Khatib 2008, p. 926) The data that might not be sent or processed in the system, can be saved to remote database and use that data to train, model or tune the control algorithms.

The components that interact with the robot can include other robots, target objects specifying the control goal, human operators, obstacles, or abstract factors such as the natural environment (Chopra, Hatanaka et al. 2015). Networked robotics is widely researched topic, especially fully autonomous robot cells in manufacturing and self-driving cars. Networked robotics challenges researchers and designers to find optimal way to control and coordinate cooperative multi-robot systems in a dynamic topology, where the robots' environment changes based on its actions. There are various issues to be solved regarding control, communication and perception – who controls who and what information is exchanged and how. (Oussama Khatib 2008, p. 985)

3.1 IoT and CPSs

IoT (Internet of Things) is a huge word today. While the word IoT is based on interconnected devices collecting and changing data, the CPSs (Cyber Physical Systems) pay more attention to how these devices cooperate between sending commands and retrieving information in difficult tasks. These are usually huge interconnected system, where every sensor, actuator and machine are working together in real time. (Luo, Chia-Wen Kuo 2016)

Real time systems pose new challenges that are limited by hardware and software. The physical world can be modeled with differential equations while the cyber world is sequential and discrete time system. These hybrid systems are continuous topic among researcher. The biggest question is: How do we control both physical and cyber world in real time, perfectly in sync, when hardware (computer chips), software layers (CPU, OS), network connections and actuators all pose lag to the system. The systems that address this problem are real-time operating systems (RTOS), which can achieve accuracy of few microseconds. (Luo, Chia-Wen Kuo 2016)

3.2 Two robot network

The goal of this thesis is to study two cooperative components that cooperate with each other in fixed topology, where the environment and robot architecture is fixed – workspace of robot cell and position is fixed, and movements are predefined by the human programmer. Taking the environment into account in the network is not our priority excluding workspace restrictions made by safety structures and the ground.

In this thesis the network is made by three components: two robot arms and the controller. The robot arms are linked to the controller and they exchange information through it. Both robots need to update their position with each other in real time to ensure smooth operation and to avoid collision. The feedback loop is formed with sensors and actuators of the robot that feed the information back and forth in a shared network. Figure 6 shows the control system of a single robot.

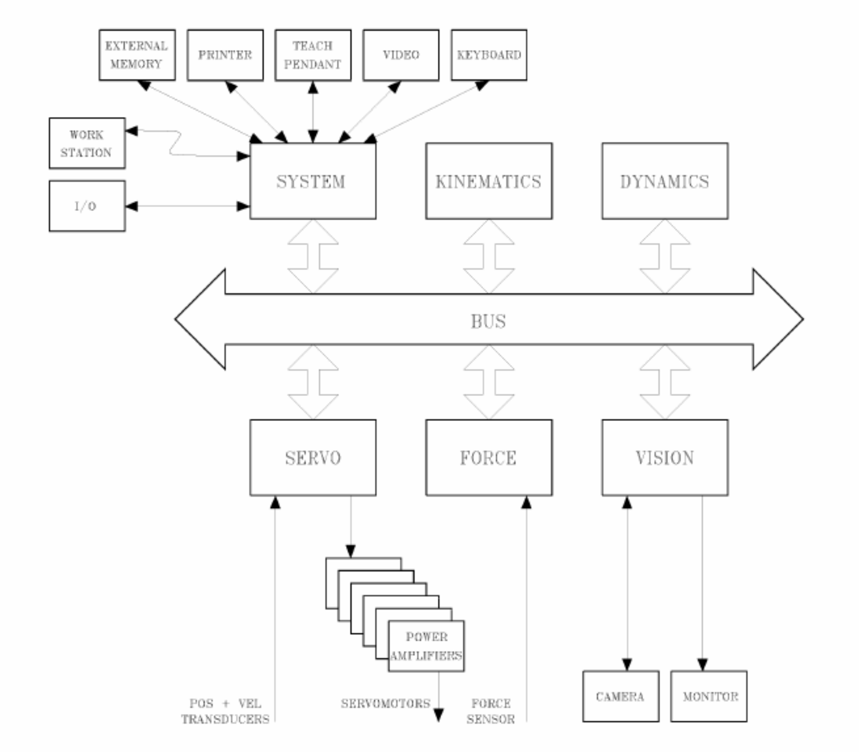


Figure 6. General model of the robot control system. (Sciavicco, Siciliano 1996, p. 330)

3.3 Sensor network

The perception of robot is based on its sensory system. The system can collect data of its internal mechanical status (proprioceptive sensors) as well as external status of the environment (exteroceptive sensors). The control system for robots combine various different engineering disciplines. It is a combination of mechanics, electronics, information and automation theory and it also includes for example: material science, signal conditioning, data processing, artificial intelligence, programming and computational architecture. (Sciavicco, Siciliano 1996, p. 2)

3.4 Machine vision

The network can also have machine vision setup, where the camera follows the workspace and, for example scans the physical location of the object and sends the coordinates to the robots handling it. This can include algorithms where the color, position, shape and location can dictate which robot in multi-robot workspace is capable to handle it. *The arguments for the request of the recognition service, depending on the recognition algorithm, can be 2-D intensity/color image, depth map, or 3-D point cloud, while the service response should contain the object type and the object pose in the environment* (Luo, Chia-Wen Kuo 2016).

This requires robust and accurate calibration of the camera's coordinate system when it is translated to robots coordinate system. Good calibration combined with CAD (Computer Aided Design) model of the object, can provide enough information (Center of mass, contact points) to the system so the proper gripping techniques can be calculated. (Luo, Chia-Wen Kuo 2016)

3.5 Multi manipulator system

Multi manipulator system has several steps that must be planned. These steps are common in manufacturing and robot cells where rigid bodies are handling a rigid object. Živanović (Živanović Vukobratović, 2006) suggests the following steps in cooperation.

1. Planning of the approach
2. Approach to the object
3. Grasping, gripping or suction
4. Lifting
5. Transferring
6. Lowering
7. Releasing
8. Withdrawing

3.6 Multi-agent architecture

Multi-agent system is made of multiple networked agents. The term agent describes an abstract subject who has a decision-making capability regarding its own state. The general goal of cooperative control is to render desirable collective behavior of multiple agents via only local interactions or communication with neighboring agents. On the other hand, the goal of cooperative estimation is to gain totally optimal estimates by efficiently collecting spatially distributed sensing data through limited information exchanges among sensors. (Chopra et al. 2015)

Luo, Chia-Wen Kuo (2016), tells that MAS is one the promising software engineering approaches for CPS problem with SoA (Service-oriented architecture).

Toshiyuki (Toshiyuki, Kazuhiro 2011, p. 111) defines the multi-agent system (MAS) by the following items:

- An environment E.
- A set of objects O located in E.
- A set of agents A, which are capable of perceive, create and modify other objects and communicate with other agents.
- A set of relations R, which relate the objects.
- A set of operations O_p, defined to allow the agents to perceive, transforms and manipulate the passive objects in O.
- A set of universal laws, that determine the consequences of O_p in the particular world.

In a two-robot arm system surrounded by security fence, there are following agents: Fence and the ground acts as an environment E, both robots are agents A, items, that the robots and handling can be seen as objects O. It is designer's job to define the set of relations R, for example speeds, accelerations, forces and distances about the objects and agents. Operations O_p can be simple operations, for example, pick-and-place.

Toshityuki also defines different requirements for a cooperative multi-agent system (MAS). The requirements must be measurable in a certain way and they must be covered one by one to effectively implement good cooperative strategy. (Toshiyuki & Kazuhiro, 2011, p. 113-114)

Deployment: this is referred as spatial distribution. The physical space is a resource and a goal at the same time. The agent's placements determine the communication strategy. Assembly robots in a closed space require different communication than mobile robots in an automated warehouse.

Multiplication: adding new members will increase the performance and efficiency of the system. New agents take more physical space so the task allocation and planning must be designed properly.

Communications: the designer must implement a communication method for agents to inform their internal states or the data that is acquired from the world.

Totipotence: A capacity of an agent to execute wide range of tasks. Totipotence is the opposite of specialization. Agents’ wide range of skills and features eliminates the downtime in case of a specialized agent’s failure. This increases the systems robustness.

Collaboration: One of the fundamental requirements in cooperation. It is necessary to distribute optimal tasks and schedules to avoid agent inactivity.

Coordination: Coordination is necessary to synchronize actions of multiple agents in effective manner and to improve the efficiency of the collaboration.

Conflict solving: Designing and implementing negotiation techniques in case of merging goals or the lack of resources, this can be the case in environment coverage, load sharing, warehouse robots or self-driving cars.

Competence: Energy optimization of agents. How can the system achieve the best profit, benefit minus cost?

Functional architecture: Agents must be able to achieve the primary goals they are assigned to reach. For example, region coverage robots must be able interpret the world with their sensorial system to be able to communicate their findings. The same principle applies for industrial robots.

World presentation: The system must incorporate mechanism to interpret data obtained from sensors. In multi-robot arm case, the system must be able to locate the tooling tip and arm positioning all the time and compare it to the environment. In dynamic environments the system can also have prediction and future states of the environment to aid the decision process.

Robustness: The main goal must be accomplished and requirements for minimum resources must be established.

Efficiency: The system must incorporate means to measure its efficiency in terms of resource wasting, time, number of agents, and balance of loads. (Toshiyuki, Kazuhiro 2011, p. 113-115)

3.7 Master/slave system

Master/slave concept is one of the earliest cooperative systems. It contains the master, which is in charge of the absolute motion of the object and the slave, which is force controlled and its behavior is based on interacting forces. (Oussama Khatib 2008, p. 711)

In a multi robot system one robot can act as a master and the rest act as slaves. The master is controlled by the controller which is operated by human operator. The slaves track the motion of the master. The robots and possible agents, like turntables communicate via a communication network.

3.8 Calibration of multi-robot systems

Calibration of robots is crucial part in robotics, especially in the medical field and high precision manufacturing. The industrial robots might not be always be the most accurate, so they need a good calibration methods for multi-robot systems related to base frame and each other. Base frame calibration, which is to determine the relative translation and rotation between base frames of coordinated robots, is a challenging and fundamental problem for coordinated multi-robot systems (Huajian Deng, Hongmin Wu et al. Dec 2015).

4 Cooperation with KUKA RoboTeam

As one example, robot manufacturer KUKA enables RoboTeam software for robot cooperation. RoboTeam and other various OEM (Original Equipment Manufacturer) software have good functional architecture, so the operator can teach the robot system at primitive level. Simple programming has various positive sides to it. It allows acquiring meaningful posture by teaching, computing the end-effectors location compared to root point, it computes servo references, motion and trajectory paths and it can be programmed to get information from external sensors. (Sciavicco, Siciliano 1996, p. 327-328)

4.1 Calibration in KUKA RoboTeam

In KUKA master/slave mode the software already knows the parameters of the robots. This makes calibration and teaching easy for the operator. First you need to calibrate the base frame coordinate system of robot relative to other and then their TCP (Tool Center Point) relative to each other. If the coordinates of the master are known, the position of slave can be attained straight from TCP and the teach pendant. (Huajian Deng et al. Dec 2015)

4.2 Calibration in KUKA system software

Before the robots can be used in geometric coupling, they must be calibrated in relation to one another. In follow mode, where slave follows the master, the participants must know the root point of the other.

The calibration tool (Ref_Pin) acts as reference point and must be calibrated on the corresponding robots. The user assigns a Cartesian coordinate system (TOOL coordinate system) to the tool mounted on the mounting flange, this is called reachable workspace of robot. The tool coordinate system (dexterous workspace) has its origin at a user-defined point (TCP) and is generally situated at the working point of the tool. Tool acts an external kinematic system and is configured in the system variable ($ETx_TPINFL) in the machine data. This contains the position of the reference point relative to the FLANGE coordinate (Figure 7). (KUKA.RoboTeam 2013)

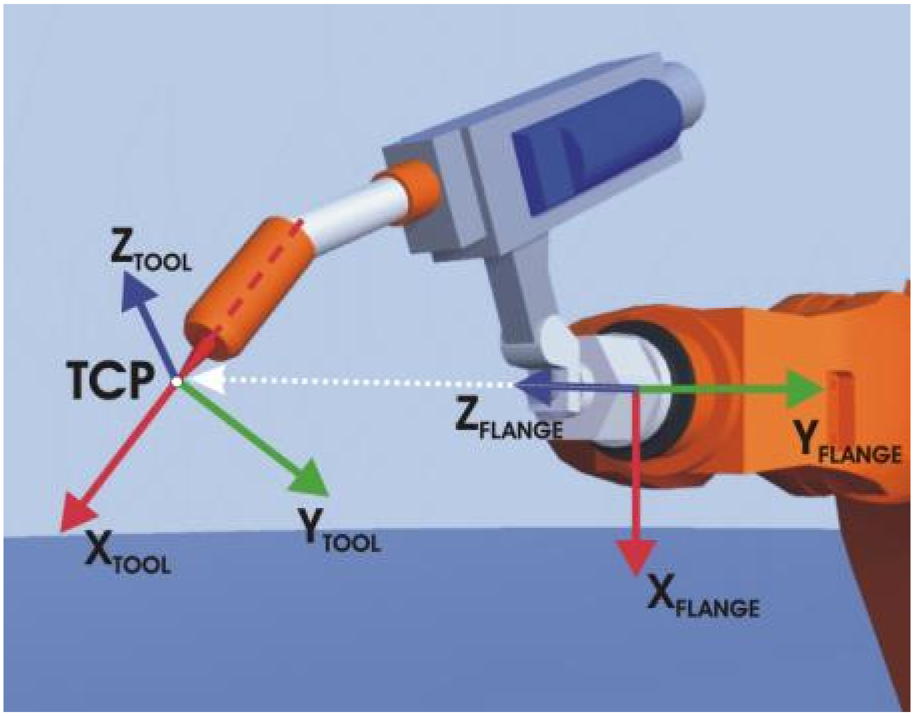


Figure 7. TCP calibration principle. (KUKA System Software, 8. 3 2013)

After the TCP is calibrated the robots are moved to reference point 4 times (Figure 8). The reference points can be freely selected, but they must be different every time and they should approximately cover the entire workspace.

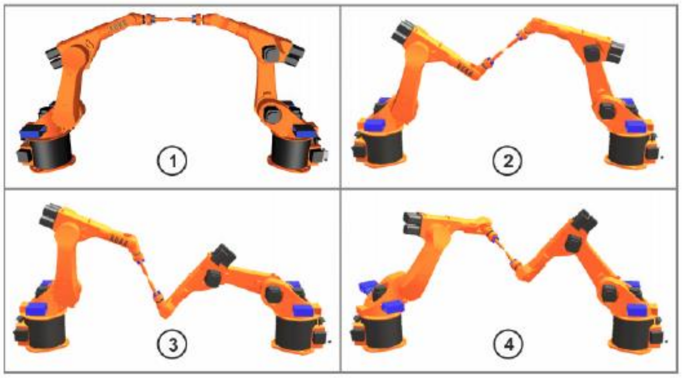


Fig. 6-4: Example: Calibration of robots relative to one another

1	1st position	3	3rd position
2	2nd position	4	4th position

Figure 8. Example of master/slave coordinate system calibration points. (KUKA.RoboTeam 2013)

4.3 Master/Slave coupling

In KUKA.RoboTeam 2.0-software the geometric coupling in master/slave-system (Figure 9) is executed the following way: The slave robot follows the flange motions of the master robot, or the flange motions of an external axis kinematic system which is connected to the master controller. The master controller cyclically transforms the current axis angles for the requested kinematic system and sends the Cartesian result frame to the slave. The slave links the kinematic evaluation with local data to the current $BASE_C (configuration base) value. Robot geometric shape is mapped to a single point in configuration base. (KUKA.RoboTeam 2013)

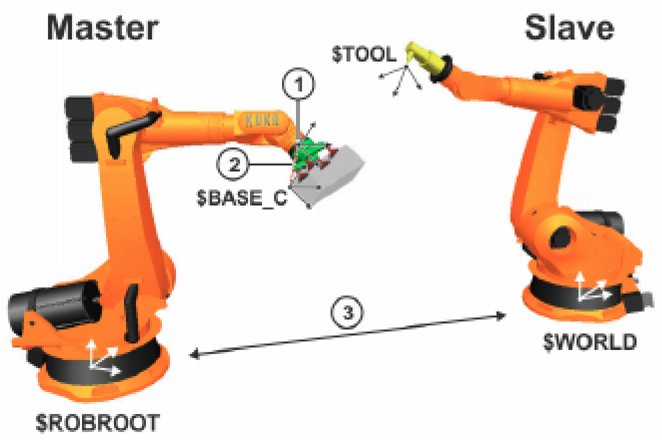


Fig. 2-3: Geometric coupling

1 Flange coordinate system
2 LK_OFFSET (difference between $BASE and the flange coordinate system)
3 LK_ROOT (difference between master $ROBROOT and slave $WORLD)

Figure 9. KUKA.RoboTeam 2.0 master/slave geometric coupling. (KUKA.RoboTeam 2013)

4.4 Load sharing

Load sharing is common in assembly robots. The robots move a workpiece together usually in a preprogrammed way, so the movements are predetermined and easy to control.

Load sharing is critical issue and from dynamic point of view, the point of interest is transfer of loads during movement. There is numerous researches and published papers about load sharing dynamics and balance issues regarding forces and moments. How to control an object simultaneously while taking the trajectory, internal and external mechanical stresses into account. The topic also covers intelligent control and fuzzy controllers in nonlinear systems.

When the manipulators hold a rigid object, then relative position and orientation are to be kept constant. When a cooperative multi-arm system is controlling a common object, it is important to control both the motion of the held object and the internal forces applied to it.

The load capacity of robots, which is usually determined by the torque limits of the actuators, is closely coupled with acceleration and speed. In assembly robots, mechanism acceleration and stiffness are often more important parameters than peak velocity or maximum load capacity. The goal is to minimize pick-and-place motions and cycle time while maintaining precision. (Oussama Khatib 2008, p. 68, 707, 711)

KUKA.RoboTeam 2.0-software has also a load sharing mode, where coupled robots move a workpiece together (Figure 10). The workpiece motions result from the programmed motions of the master and the slaves follow the flange position of the master.

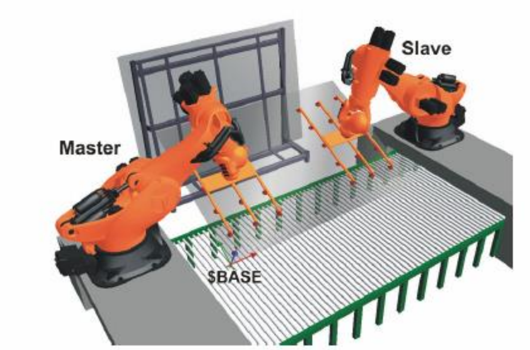


Figure 10. Robot load sharing. (KUKA.RoboTeam 2013)

5 SUMMARY

Multi-robot cooperation combines almost every technical field. This makes it challenging and ever-evolving field of research. Continuous research has showed various different methods and approaches to solve the problems that today’s industry poses.

While designing multi-robot systems, one has to consider the requirements for robots. Depending on requirements the robot teams can consist of heterogeneous or homogeneous teams and both have their own strengths. This thesis revealed many different questions that must be answered. What are the goals and tasks of robot teams, what kind of operation is required, what kind of objects the robots are handling and how multi-robot systems are calibrated. This requires knowledge of physical constraints like kinematics, internal forces, sensor placement and the operational workspace of the robot.

Hardware level of robots is well known and documented. The main study of multi-robot systems is their control. New technology has speeded up the manufacturing, but it has also brought a set of new challenges, for example, how machine vision can detect objects and drive actuators of robot within its’ physical constraints or how two robots can handle the same object at the same time.

The systems also need a software. The software requires fast algorithms, robust safety systems, sensor systems and communication network. The software will always have lag, and this can be solved with real-time operating systems.

The goal of the equipment manufacturers is to make the use their software simple and KUKA has made it easy for operators to operate and calibrate robot systems and they have their own system software, RoboTeam for multi-robot cooperation. RoboTeam can pair robots as master and slave and, for example, use them in load sharing applications.

6 REFERENCES

AKELLA, S. and JUFENG PENG, 2004. Time-scaled coordination of multiple manipulators, 2004, IEEE, pp. 3344 Vol.4.

CHOPRA, N., HATANAKA, T., FUJITA, M. and SPONG, M.W., 2015. Passivity-based control and estimation in networked robotics. Springer.

DAVID J BLAND, Oct 17, 2017-last update, The Business Model for Robot Food. Available: https://medium.com/precoil/the-business-model-for-robot-food-136d461fbfb0 [cited 04.05.2019].

FARINELLI, A., IOCCHI, L. and NARDI, D., 2004. Multirobot systems: a classification focused on coordination. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), **34**(5), pp. 2015-2028.

GERAVAND, M., FLACCO, F. and DE LUCA, A., May 2013. Human-robot physical interaction and collaboration using an industrial robot with a closed control architecture, May 2013, IEEE, pp. 4000-4007.

GUDLA, A.G., 2012. A methodology to determine the functional workspace of a 6R robot using forward kinematics and geometrical methods, ProQuest Dissertations Publishing.

HUAJIAN DENG, HONGMIN WU, CAO YANG, YISHENG GUAN, HONG ZHANG and JIANLING LIU, Dec 2015. Base frame calibration for multi-robot coordinated systems, Dec 2015, IEEE, pp. 1489-1494.

KUKA SYSTEM SOFTWARE, 8. 3, 22 April, 2013-last update, KUKA System Software [Homepage of KUKA Roboter GmbH], [Online]. Available: https://www.kuka.com/fi-fi/products/robotics-systems/software/system-software/kuka_systemsoftware.

KUKA.ROBOTEAM, 2.0., 30 April, 2013-last update, KUKA.RoboTeam 2.0. Available: https://www.kuka.com/fi-fi/products/robotics-systems/software/hub-technologies/kuka_roboteam.

L.E. PARKER, 1998. ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation. IEEE.

LUO, R.C. and CHIA-WEN KUO, 2016. Intelligent Seven-DoF Robot With Dynamic Obstacle Avoidance and 3-D Object Recognition for Industrial Cyber-Physical Systems in Manufacturing Automation. Proceedings of the IEEE, **104**(5), pp. 1102-1113.

OUSSAMA KHATIB, 2008. Springer Handbook Of Robotics. 1 edn. Berlin: Springer-Verlag.

SCIAVICCO and SICILIANO, 1996. Modelling and Control of Robot Manipulators. 2 edn. Springer-Verlag London.

TOSHIYUKI, Y. and KAZUHIRO, O., 2011. Multi-Robot Systems, Trends and Development. 1 edn. Rijeka, Croatia: InTech.

ŽIVANOVIĆ, M.D. and VUKOBRATOVIĆ, M., 2006-last update, Multi-arm cooperating robots [Homepage of Springer], [Online document]. Available: https://doi.org/10.1007/1-4020-4269-8.