POLITEHNICA UNIVERSITY OF TIMISOARA
ELECTRONICS AND TELECOMMUNICATIONS
FACULTY

Mathematical analysis and optimisation of offline programmed robot cells in industrial applications

M. Sc. Dipl.-Ing. (FH) Michael Kleinkes

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electronics and
Telecommunication Engineering
in the Faculty of Electronics and Telecommunication of the
"Politehnica University" of Timisoara
Timisoara, 2009

Advisor:
Prof. Dr. ing. Alimpie Ignea

Abstract

This thesis is dealing about correction and improvement of modern industrial robot applications. Focussing on the non-linear correction of robot linear tracks, it is minimizing the 6-dimensional error of the reached robot points in applications. Due to the compensation of the gap between simulated robot cell and real application, the output of offline robot program points without the need for online correction is significantly increased. The improvement is mainly reached by the continuous mathematical description of the practical linear track geometry.

Acknowledgements

I would like to thank my supervisor Prof. Dr. Alimpie Ignea for his professional and always kind support during this thesis. His helping comments and his patient and flexible cooperation was giving me the needed space for combining PhD-thesis and daily work together. The impressing meticulous correction of rough versions of the thesis and the gracious way of writing always respectful criticism, was encouraging me during long working nights.

Also I would like to thank my friend Dr. Werner Neddermeyer for always supporting me and for giving me a guide for life. I hope that I can reach one day his level of engineering brilliancy and character magnitude.
For their encouragement and support, I want to thank also Prof. Dr. Wolfgang Winkler, Dr. Adrian Krzizok and Dr. Liviu Toma.
A special thank to my friend Dr. Anna Lilienthal, who was setting up the foundations to this work and who teached me in long sessions the basics of robotics during my student phase at university.
All my friends I am thanking for reading and correcting my documents and for the understanding of my time-consuming "hobby" in the last years. Even my parents and family, which were always encouraging me, even in bad times.

Most of all, I am thanking my girlfriend Sarah for all lifetime, she invested for me while abstaining from common weekends or nights. Her appreciation and love for me are not measurable and are the most precious present, I ever got.

Contents

List of Figures

List of Tables

List of Abbreviations

AProDoR *Automated program for documentation of robot movements*
ABB IRB 2400 *Robot model from swedish robot manufacturer*
CAD *Computer aided design*
DFT *Discrete Fourier transformation*
DHP *Denavit-Hartenberg parameter*
FFT *Fast Fourier transformation*
GUI *Graphical user interface*
ISO *International standardization organization*
KRL *Programming language for KUKA robotis*
LTD800 *Model of Leica laser tracker*
MFC *Microsoft foundation class*
MTBF *Mean time before failure*
NIST *National institute for standards and technology*
PC *Personal computer*
PDF *Portable document format*
RAPID *Programming language for ABB robots*
TCP *tool center point*
TTL *Transistor-transistor logic*

1 Introduction

In modern industrial robot applications, the demands for flexible automation are increasing with the growing span of technical solutions which are implemented today. Intelligent sensing, realized by vision systems, complex force feedback sensors or any other mechanisms enables the robot system to adapt itself to changing situations and tasks.

Proportional to the increased intelligence and more important to the increased complexity, the effort for setting up the robot system is also getting more. Not considered in many industrial applications today, a proper setup of the robot with integrated calibration of its peripherals is providing the needed quality of the tasks which are executed by the robot.

Influences of peripheral components will take crucial effect on the resulting quality performance, even if highly accurate robots are used in the application. Positional and rotational influences caused by non-calibrated linear track or incorrect determined robot tool data can be avoided with a minimum effort in time and money.

The result of this thesis is a decreasing of the effort for creation of a successful working robot application and besides this also increasing the resulting quality of fulfilled tasks. It gives an overview about the common problems occurring during setup and explains effects and influences of various parts of the robot system.

The focussed main goal of this thesis, which is reached by new methods for measuring and modeling of the linear track, is the reduction of the deviation between CAD model and real behavior of the robot system.

This thesis is minimizing the gap between offline and online world in modern robotics under main focus to used linear tracks. This is an important contribution and an improvement of the common setup process for industrial robot application resulting in cost and time savings.

2 Background and related work

The current state of the art in automotive sector of industrial robotics is focussed on robot program construction under usage of CAD-systems. A successful automation of the robot program creation is linked to several conditions to the setup of the application and required exactness.

This thesis is combining multiple different fields of investigation concerning robot applications. Relating to this, existing methods and systems were extended and optimized during integration process for the creation of a strategy for online correction minimization.

For the first step, the analyzation of the robot system, [3] is building a basic part of my investigations and contributions. The work deals with the automated evaluation of robot programs and was the initiator for this thesis. It shows methods for efficient analyzation of static and also dynamic robot movements, everything together integrated in a software module.

The mentioned linear track analysis was started in [27]. It is refined by the automated correction, mentioned in [25] and a description of the method explained in subchapter 4.3.

Concerning the problem of copy and mirror robot programs, [24] is giving a detailed overview about the problematic. It has also an interesting approach about needed tool configurations for an successful mirroring.

Further interesting steps for the analyzation of the error propagation can be found in [26]. The implemented simulation of various parts of the robot cell has given an useful entry into the accuracy analyzation of the application.

All mentioned documents were supported by the author in pre-phase of this thesis. They were planned to build a foundation for the contributions which are made in this work.

3 Robot

3.1 Introduction

An upcoming dispute about the question *"What is a robot?"* was the reason of an arguing discussion in the crowded hall of the St. Martin theater in London. Initiator was the appearance of the drama "Rossums Universal Robot (R.U.R)" [1] by Karel Čapek (1890 - 1938). The questioner gave himself the answer later: ".... A being without individuality and initiative, which has to fulfill every command given to him".

Based on the increasing flexible automation, today robots are used in different varieties of applications. They help to accelerate, improve, secure or automate working processes. Famous example applications of robots are checking tubes, secure bombs, help physically disadvantaged people or construct houses. This is only a tiny facet of the whole possible field of applications.

This thesis is dealing with industrial robots, which are commonly used in automotive industry. Application tasks for such robots, for example, are point welding, sealing or part handling. To fulfill the first law of robotics, that the robot is *exactly* doing what he was programmed, in this work a small part is mentioned, out of the various aspects which have to be taken in consideration - *robot movement accuracy.*

For respecting this law of robotics, exact movement like programmed, the movement exactness of the robot has to be exactly the same in reality and theory. In reality, this perfect movement is influenced by many different parts of the robot system. To achieve in real applications a compromise to the impossible high accuracy, the needed accuracy to fulfill the application task is analyzed and the robot adjustments adapted to them. The robot is always expected to be as accurate as possible, but for practical application it is sufficient if the demanded accuracies are reached. This demands can extremely vary, depending on the different application needs.

For an analysis of the robots accuracy, the robots tool is measured in static points and movements. Measurement of the robot means in this case a 3-dimensional detection of a fixed point on the robot using an appropriate measurement system. This fixed point for the accuracy analyses is the robots tool center point (TCP).

For the determination of the robots accuracy, the measured 3-dimensional positions of the TCP can be compared to the programmed points in the robot control. Based on an european standard for the determination of robot accuracies [2], the key parameters to describe the resulting exactness in the robot movements can be calculated.

To simplify this calculation for the user, a software (section 4.8) was implemented during this work. Taking the measured robot TCP positions and movements during the accuracy test, it calculates the accuracy parameter of the robot automatically. Additionally, diagrams for a quick overview or usable for error component identification are generated and displayed. The software is presented in section 5.1 followed by some example analysis results.

3.2 Static robot accuracy

For many applications today, only the static accuracy of the robot is important. For tasks like point welding on a car body, it is needed that the robot is moving exactly to a programmed position. Otherwise, the remaining deviation between teached position and the real reached position of the robots TCP is the inexactness of the welding position.

For TCP movement trajectory planning, a set of movement and control commands - the robot program - has to be created and provided to the robot control. This program is written in a robot-specific language (for example KRL or RAPID) and can be created in an *offline* or in an *online* way of programming [70].

The basic way of programming the robot is the *online programming mode*. In this case, the robot is moved to its final application positions and the corresponding coordinates of the TCP are stored into the robot program. The exact coordinates for each so-called "teached" position of the TCP are delivered by the joint angle encoders of the robot.

Online robot programming offers an easy compensation of all occurring error influences to the robot system. Implicit compensation by teaching desired TCP positions manually is a time-consuming, but exact way of programming. The resulting accuracy during application run is then only influenced by the robot ability to move exactly to the same teached point again, the robots *repeatability*.

Besides of this accuracy advantage, the online programming of an industrial robot is combined with much human effort. The robot has to be moved by hand exactly to every application point and this point has to be stored into the robot program. This time-consuming procedure has, during increasing complexity of the applications, brought new ideas how to improve the programming process.

Using special simulation programs for virtual emulation of the robots movements, nowadays robot programs can be created in an *offline programming mode*. The user can move the robot virtually in CAD environment to an application point and store the TCP position in a robot program. Into the simulation program, all information about the CAD positions of the application cell components - especially the working objects - are integrated.

The program mode is called offline, because there is no need for the user, to stay at the robot for teaching the application points. The programs can just be programmed in office, while the robot application can be used in application. This gives a huge amount of effectiveness and time savings to the programming process and is a common used method today in automotive industry.

A crucial weakness point of the online programming mode is the fact, that the simulation software is calculating all robot TCP positions based on a set of CAD data and - most important - on the fact, that the robot is perfectly moving like the simulation model of the robot. The real movement of the robot TCP will always differ from this theoretical model, due to the robots inaccuracy. Depending on the robots accuracy, the resulting program is more or less usable in an application. A 100%-usage of the offline teached robot programs is not reached yet in high accuracy applications. In common case, the user has to adapt a subset of the online programmed points. But this subset is only a percentage of all application points and can be corrected with reasonable effort.

3.2.1 ISO 9283

This international standard describes, how to test an industrial robot to identify its static and dynamic accuracy. A 6-dimensional pose accuracy in the static case of a robots TCP is divided into positional and repeating accuracy. The two different parameters can be calculated on measurement data of the same test run, but are calculated in different ways.

Positional accuracy

The deviation between a programmed robot position and its real final movement position after swinging, is described in the standard as positional accuracy. This deviation is calculated by using the mean value of deviations during a special test run with moving multiple times to one test point. The final positional accuracy is defined by the deviation between the center-of-gravity of this test runs and the programmed position. The orientation accuracy is calculated similary.

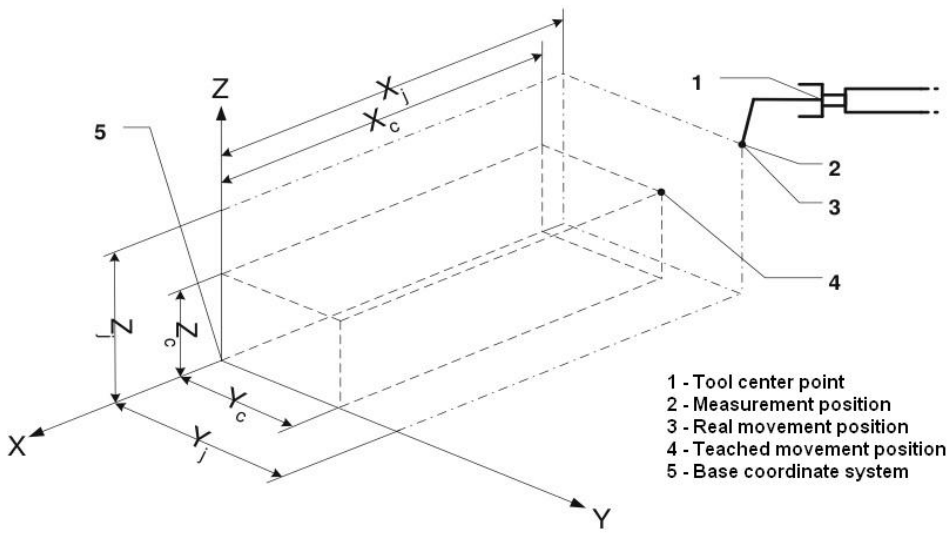

Figure 1: *Relation between teached and measured TCP position (taken from [2])*

Figure 1 displays the different positions during the test run for determination of the static accuracy. The programmed robot TCP position, stored in the robot program and processed by the robot control can be found as point "4" and is described by the coordinates x_c, y_c and z_c. The measured positions are described by point "3" and uses the coordinates x_j, y_j and z_j.

The absolute positional deviation AP_p can now be calculated using the corresponding deviations for each single direction by:

$$AP_p = \sqrt{AP_x^2 + AP_y^2 + AP_z^2} \tag{1}$$

To get the single direction deviations, the difference between teached position and the mean of the measured positions is calculated by:

$$AP_x = (\bar{x} - x_c), \qquad AP_y = (\bar{y} - y_c), \qquad AP_z = (\bar{z} - z_c) \tag{2}$$

using

$$\bar{x} = \frac{1}{n}\sum_{j=1}^{n} x_j, \qquad \bar{y} = \frac{1}{n}\sum_{j=1}^{n} y_j, \qquad \bar{z} = \frac{1}{n}\sum_{j=1}^{n} z_j \tag{3}$$

so that the point defined by $\bar{x}$, $\bar{y}$ and $\bar{z}$ is the center of gravity for the n measured points during the test move.

The missing accuracy parameter to describe the 6-dimesional static accuracy is given by calculation of the orientation devitation. The calculation of the differences between given and measured orientation is equal to the procedure for calculation of AP_p:

$$AP_a = (\bar{a} - a_c), \qquad AP_b = (\bar{b} - b_c), \qquad AP_c = (\bar{c} - c_c) \tag{4}$$

where a, b and c are representing the three different angles with:

$$\bar{a} = \frac{1}{n}\sum_{j=1}^{n} a_j, \qquad \bar{b} = \frac{1}{n}\sum_{j=1}^{n} b_j, \qquad \bar{c} = \frac{1}{n}\sum_{j=1}^{n} c_j \tag{5}$$

Different to AP_p, the single orientation deviations are not combined together. The absolute static positional accuracy is represented by AP_p and the three orientation parameters AP_a, AP_b and AP_c.

Repeating accuracy

Once determined the static positional accuracy, it is possible to make predictions about how accurate the robot is able to move to one given TCP position. Like mentioned before, this is an important accuracy parameter for offline programming.

In case of online programming, the robot should only be capable to move to a prior teached position again. The repeating accuracy of the robot describes this, using four different parametes: The positional repeating accuracy RP_l and the orientation accuracy departed into three parameters RP_a, RP_b and RP_c.

To reach a prior TCP position again is easier for the robot because the robots repeating accuracy is normally 5 to 10 times better than the absolute accuracy.

An identification of the repeating accuracy can be done based on the measurement results of the static accuracy test run with slightly adapting the calculation. The parameter RP_l defines the 3σ-intervall and with this the radius of a sphere, in which 99.7% of the measured positions are integrated (see equation 7). It is won out of the mean value $\bar{l}$ of the single deviations l_j:

$$\bar{l} = \frac{1}{n}\sum_{j=1}^{n} l_j \quad \text{with} \quad l_j = \sqrt{(x_j - \bar{x})^2 + (y_j - \bar{y})^2 + (z_j - \bar{z})^2} \tag{6}$$

Slightly different to the positional accuracy, the final repeating accuracy is given by RP_l as the radius of the deviation ball and a triple standard deviation of the positional deflections:

$$RP_l = \bar{l} + 3 \cdot S_l \tag{7}$$

using

$$S_l = \sqrt{\frac{\sum_{j=1}^{n}(l_j - \bar{l})^2}{(n-1)}} \tag{8}$$

The corresponding orientation repeating accuracy is calculated by getting the standard deviations of the measured orientations to their mean value in every single degree of freedom. With this, even for the orientation repeating accuracy there are three different parameters:

$$RP_a = \pm 3S_a = \pm 3\sqrt{\frac{\sum_{j=1}^{n}(a_j - \bar{a})^2}{(n-1)}} \tag{9}$$

$$RP_b = \pm 3S_b = \pm 3\sqrt{\frac{\sum_{j=1}^{n}(b_j - \bar{b})^2}{(n-1)}} \tag{10}$$

$$RP_c = \pm 3S_c = \pm 3\sqrt{\frac{\sum_{j=1}^{n}(c_j - \bar{c})^2}{(n-1)}} \tag{11}$$

3.3 Influences on the static accuracy

The possible influences of the robot, which are decreasing the static accuracy can basically be divided into two different parts. On the one hand, there is a group of influences, which are formed by model parameter deviation. In case of inaccurate geometrical parameters, the transformations for position calculation can not be calculated exactly. This results in a wrong or at least not accurate calculation of the robots TCP position.

All influences, which are deviated from the theoretical set of parameters are bunched together to this so-called primary error influences. The primary error influences can be avoided by an appropriate identification method, mentioned in [5].

On the other hand, there are the secondary error influences. These are all non-deterministic error components and even all components, which can not be identified by [5]. In the following parts of the chapter, some example error influences are mentioned.

3.3.1 Primary influences

The sorting of error influences into primary and secondary bases on the used algorithm for identification. In this thesis the algorithm mentioned in [64] is used and all following sortings are based on this.

Robot base frame

For offline programming of industrial robots it is needed to identify the robot base frame in world frame coordinates. This identification can be done in different ways, using more or less accurate measurement devices, or even the robot itself.

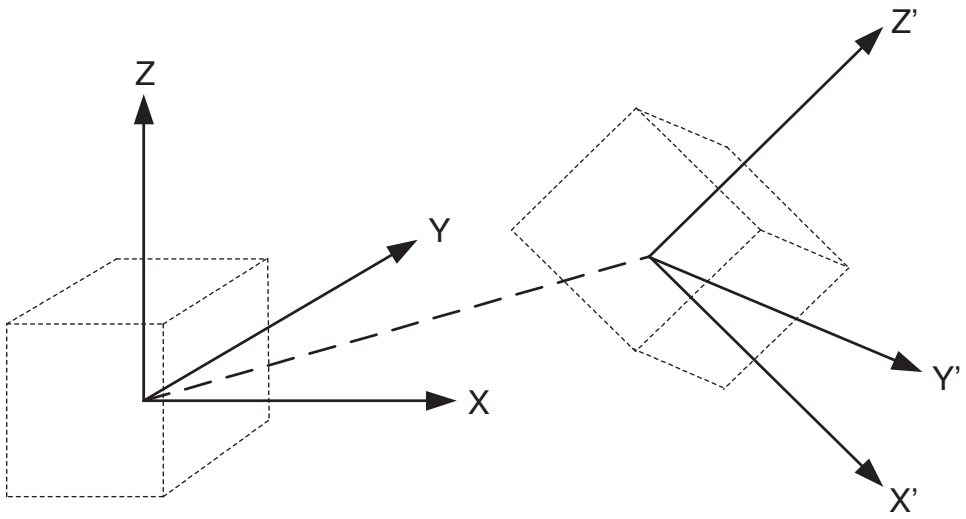

Figure 2: *Error in robot base frame identification*

Deviations in identification of the robot base frame are linear errors, which are getting higher with TCP-positions far from the robot base. For example, an deviation of 0.5 degree after base frame identification results in a positional error of several millimeter.

How accurate the base frame can be identified and how many measurement positions have to be used for identification is explained in [8].

Dimension errors

For getting the current TCP-position of the end effector, the robot calculates its reverse transformation. For this, the current values of the joint angle encoders are read and processed with a set of Denavit-Hartenberg-Parameters (DHP) [5].

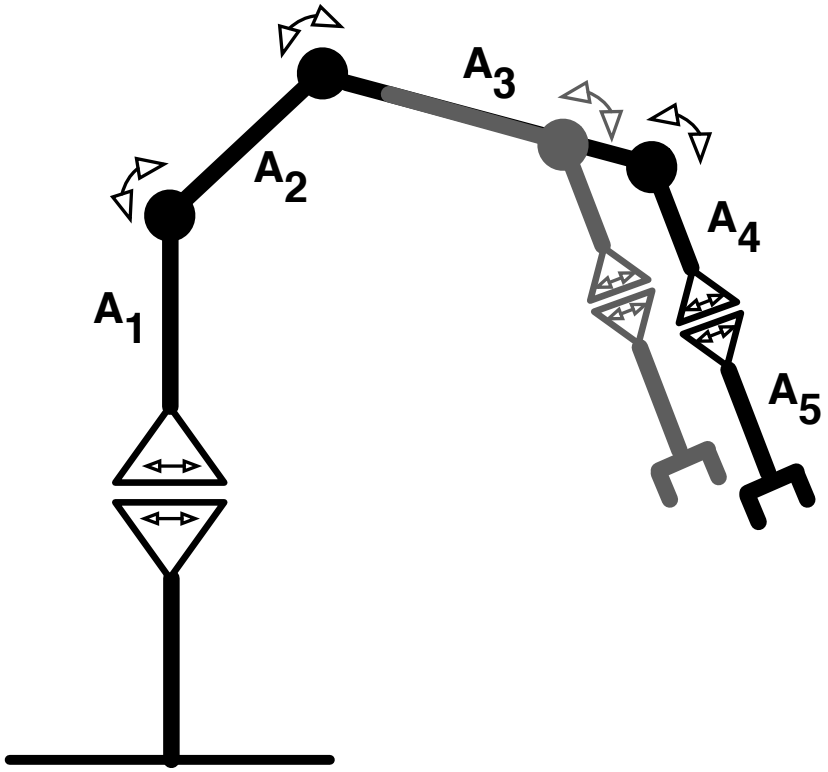

Figure 3: *Dimension error in* A_3

For an exact calculation of the current TCP position the DHP have to be as perfectly fitting to the real robot model as possible. Generally, these parameters are adapted to the robots theoretical CAD-model with differences in the size and angles of the used parts of the real model. A post identification of the DHP (see [9]) can be done using a high-accurate measurement system to minimize the dimension errors.

Elasticity of kinematic chains

During robot arm movements, a difficult composition of forces is taking its influence on the joint motors. Due to a limited stiffness of the robot arm components, the TCP positions which were planned by the robot control theoretically, are influenced practically by deformations of the robot arm. Dependent on the current TCP position and tool weight, these deformations are taking more or less part to the resulting robot accuracy.

In case of TCP positions with a long absolute vector in robot base frame, so at all points with long distance from the robot base origin, the arm has to be stretched out to reach the position. In consequence, the gravity lever working on the joints - respectively on the motors and gears - is increased by the joint angle position.

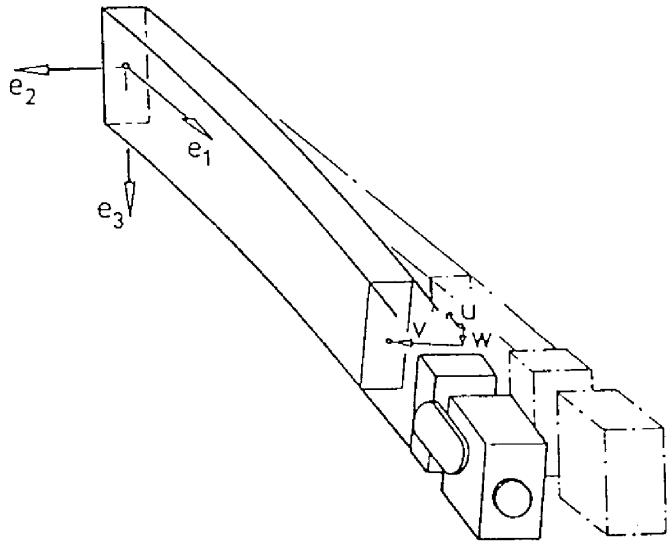

Figure 4: *Positional deflection caused by flexibilities*

Increasing moments and forces on the metal parts of the robot arm are causing deformations based on the part flexibilities. With that, the deformation of the robot arm components is depending on the current arm position and even increased during movement positions at the outer zone of the working range of the robot.

3.3.2 Secondary influences

As already mentioned, every error influence part, which can not be identified by the chosen method for parameter identification, is categorised in this thesis into the secondary influences on the robots static accuracy. Integrated into this category, there are also non-deterministic error influences.

Sensor and processing errors

For one movement of the robots TCP to a dedicated 6-dimensional position in its working area, the robot control has to calculate the appropriate movements of every single joint of the robot. Like explained in chapter 3.2, the real movement of the robot TCP is never like the theoretical movement calculated by the robot control, because of several error influences.

To construct a closed loop control for the position of the robots TCP, the robot control has to get a feedback signal of the angle sensor of every robot joint. These angle encoders can be realized in different sensor types like resolver, incremental optical or absolute optical encoders for example.

Each of the sensor systems has got its advantages and for sure even disadvantages. As a basic fact, every angle encoder will have got a limited measurement resolution. This inexactness in measurement of the current angle is even forced by a clocked output timing. Working with a fixed sampling rate, the robot control will request the angle values every n ms from the encoders.

During the closed loop control of the TCP position while moving, the angles of the single joints can on the one hand be only resoluted in a limited way, and on the other hand only processed with a fixed sampling rate. In a theoretic optimal case, in which all other robot influences could be eliminated, the resulting robot accuracy would still be dominated by the error during sensoring and discrete processing of the sensor results by the robot control.

Gear loss

In general, the gears of the robot arm joints are stressed with two different main error components. The flexibility of the used material of the gear wheels has got a flexibility which causes deflections proportional to the adjacent forces. This was mentioned in subchapter 3.3.1.

The second occuring effect are the gear losses. For getting a compromise between longlivity and accuracy, every gear wheel has got a small gap to its neighbor gear wheel in the gear box. This gap is needed for a compensation of the tensions during movements of the robot arm.

A very small gap between two gear wheels would cause in much friction during its working, increased by the moments and forces on the gear wheels. With increasing friction, the material lost is even increased and the mean time before failure (MTBF) of the whole gear box is decreased.

With an increasing of the gap size, the accuracy of the gear box is getting worse due to the "tooth jump"-effect. If one joint has to change its movement direction during a TCP movement, the direction of the force on the tooth on the gear wheels is also flipped. But caused by the small gap between the gear tooth, a flipping of the movement direction of the gear wheel also includes a bridging of the gap.

To get from one gear tooth to the next gear tooth during direction flipping, the gap has to be "jumped". This jump is a small movement of the gear wheel with no effect to the robots TCP. The robot control is calculating a rotation angle for the joint, which does not result in the assumed position. Due to this fact, the gear losses are causing an error component to the accuracy of the robot system.

Temperature drift

Every deviation of the real robot from its theoretical model integrated in the robot control is causing differences between real and calculated robot TCP position. An exact set of geometrical parameters for the robot is a base for a proper calculation of the robots transformations [9].

Once adapted the geometrical parameters of the robot to its real model, the resulting accuracy is only constant, if assumed that the robot parts are not changing in size. But during changes of the outside temperature of the robots application place, the used parts are underlying an extension or contraction.

These are deviations from the calculated geometrical parameters, resulting in the already mentioned positional problems. Due to a temperature compensation of this parameters, adapted to the environment temperature, this error components can widely be eliminated.

Overall, there are multiple influences to the robots accuracy, caused by the various components and systems of a robot application system. The influences mentioned in this thesis are to give a rough overview about critical parts of the complicated system of different error influences.

3.4 Dynamic robot accuracy

Today many robot applications are focussing on the dynamic behaviors of industrial robots [69]. Moving a programmed path, it is important for the user, how quickly the robot can process its generated program for fulfilling his task.

With every increasing of the applied movement speed during application, the time for one workpiece to apply is decreasing. The created output of applied workpieces per hour is a value to measure the efficiency of the robot application.

Important during the increasing of the movement speed, is a compromise between velocity and quality of the application. Using an increased application speed, the error influences on the robot are increasing also and the resulting accuracy will go down to a level, which will be insufficient for the application.

To implement the requested complex movement trajectories of todays applications, simple point-to-point movement commands are not longer appli-

cable. Today modern robots provide a wide set of different instructions to adapt the complexity of the robot programs to the needed demands of the application.

Special movement instructions for programming of geometrical shapes like semi-circles or elliptic shaped curves for example, are nowadays included into the standard robot instruction set. Using this instruction sets, nearly all complex movement geometries can be programmed with a justifiable effort in time.

But the real moving of the TCP depends on the used speed more or less deviating from the programmed geometries. As a simple example for this behavior, there is an easy-to-implement edge movement. A simply composition of two linear movements, perpendicular hitting each other, is in reality not possible for the robot to move along this programmed path (see section 3.4.2).

Different effects are even here influencing the robot [67], [68], [71]. One of the most crucial one is the increasing mass inertia during high speed movements. Like the static accuracy, it is even appointed by the ISO 9283 how to test a robots dynamic accuracy, to get qualitative predictions about its dynamic behavior.

3.4.1 ISO 9283

Like in static case, the ISO 9283 defines two different indicators, which describe the dynamic accuracy of industrial robots. The dynamic influences during robot movements are depending on several predictions like for example movement speed, applied tool weight and moved geometry. The definition, how accurate an industrial robot is able to move along a given trajectory, is given by the parameter AT in the ISO 9283. More exactly it is the definition how accurate the TCP of the robot can be moved n-times along the same path in the same direction.

Two different factors together are representing the dynamic accuracy: the positional path accuracy AT_p and the composed orientation path accuracy AT_a, AT_b and AT_c. The positional path accuracy AT_p represents the deviations between the given line and the center-of-gravity of all measure lines, the orientation path accuracy equally for orientations.

In figure 5 it is illustrated, how the positional path accuracy AT_p can be calculated. "3" is showing the programmed trajectory, which the robots TCP is supposed to move on. The center-of-gravity line of the measured movements is shown in "1", which can be seen is shifted by AT_{pi}.

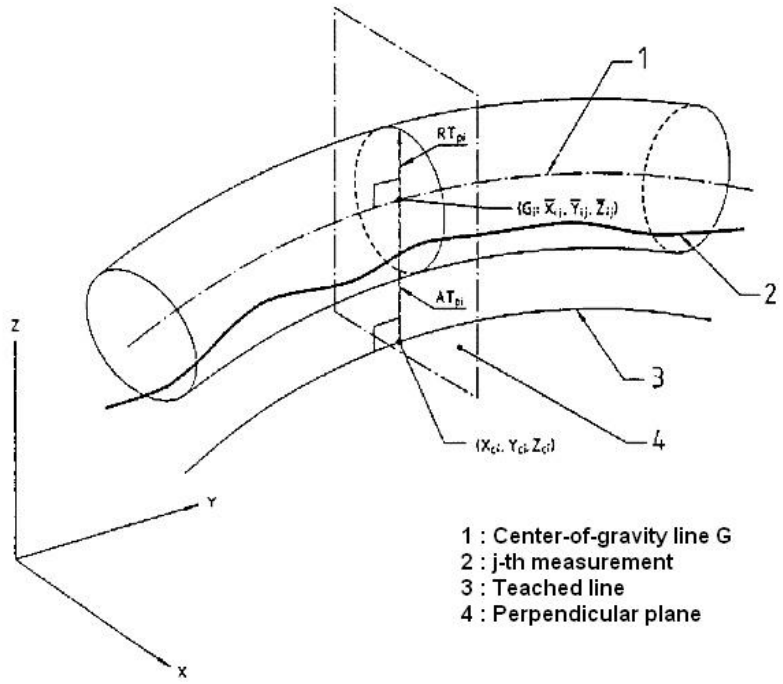

Figure 5: *Dynamic path accuracy (taken from [2])*

One sample measured movement is displayed by "2", which line is situated in a tube around the teached path "3". This tube is constructed using the path repetition accuracy RT_{pi}, explained later in this chapter. The deviations of the measured TCP positions to the teached trajectory are always calculated perpendicular to "3", as can be seen by "4". The mean value of the real movement paths is depicted by line "1".

It can be calculated by:

$$AT_{pi} = \sqrt{(\bar{x_i} - x_{ci})^2 + (\bar{y_i} - y_{ci})^2 + (\bar{z_i} - z_{ci})^2}, \qquad i = 1 \dots m \tag{12}$$

with

$$\bar{x_i} = \frac{1}{n}\sum_{j=1}^{n} x_i \quad \bar{y_i} = \frac{1}{n}\sum_{j=1}^{n} y_i \quad \bar{z_i} = \frac{1}{n}\sum_{j=1}^{n} z_i \tag{13}$$

The coordinates $\bar{x_i}$, $\bar{y_i}$ and $\bar{z_i}$ are representing the center-of-gravity line of the measurements. The appropriate point on the teached line is described by x_{ci}, y_{ci} and z_{ci}, to which the perpendicular plane was constructed.

The accuracy during dynamic processes is also expressed in a 6-dimensional way. Additionally to the accuracy in x, y and z, the orientation path accuracy can be expressed by three components AT_a, AT_b and AT_c. These are expressing the maximum orientation deviation from the teached trajectory in each single orientation direction:

$$AT_a = max|\bar{a_i} - a_{ci}|, \qquad i = 1 \dots m \tag{14}$$
$$AT_b = max|\bar{b_i} - b_{ci}|, \qquad i = 1 \dots m \tag{15}$$
$$AT_c = max|\bar{c_i} - c_{ci}|, \qquad i = 1 \dots m \tag{16}$$

with

$$\bar{a_i} = \frac{1}{n}\sum_{j=1}^{n} a_{ij} \quad \bar{b_i} = \frac{1}{n}\sum_{j=1}^{n} b_{ij} \quad \bar{c_i} = \frac{1}{n}\sum_{j=1}^{n} c_{ij} \tag{17}$$

The teached orientations in a single point (x_{ci}, y_{ci}, z_{ci}) are described by a_{ci}, b_{ci} and c_{ci}. $\bar{a_i}$, $\bar{b_i}$ and $\bar{c_i}$ are representing the mean values of the measured orientations.
As shown in figure 5, the tube around the teached trajectory is defined by the path repeating accuracy. Even equal to the static case, it can be described as the exactness, which can be achieved by the robot to repeat a special movement.

In many applications the path repeating accuracy RT_p (see figure 5) is used to classify the robot. A teached movement is optimized in practical application and then only repeated by the robot. If the online programming method was used for construction of the robot program, RT_p is an important factor for the application accuracy.

The radius of the tube spanning around the teached path related to ISO 9283 is the zone, in which 99,7% of the measured points are located. It is the deviation of the teached trajectory to the center-of-gravity line of the measured trajectory, including a triple standard deviation of Gaussian distribution:

$$RT_p = \max RT_{pi} = \max[\bar{l_i} + 3S_{li}], \qquad i = l \ldots m \tag{18}$$

with

$$\bar{l_i} = \frac{1}{n}\sum_{j=1}^{n} l_{ij} \quad \text{and} \quad l_{ij} = \sqrt{(x_{ij} - \bar{x_i})^2 + (y_{ij} - \bar{y_i})^2 + (z_{ij} - \bar{z_i})^2} \tag{19}$$

The standard deviation can be calculated in common way:

$$S_{li} = \sqrt{\frac{\sum_{j=1}^{n}(l_{ij} - \bar{l_i})^2}{n-1}} \tag{20}$$

With that, the orientation repeating path accuracy is constructed and calculated equally to the prior mentioned, out of the maximum of the differences of the single orientations:

$$RT_a = max\left(3 \cdot \sqrt{\frac{\sum_{j=1}^{n}(a_{ij} - \bar{a_i})^2}{n-1}}\right) \tag{21}$$

$$RT_b = max\left(3 \cdot \sqrt{\frac{\sum_{j=1}^{n}(b_{ij} - \bar{b_i})^2}{n-1}}\right) \tag{22}$$

$$RT_c = max\left(3 \cdot \sqrt{\frac{\sum_{j=1}^{n}(c_{ij} - \bar{c_i})^2}{n-1}}\right) \tag{23}$$

3.4.2 Dynamic behavior and influences

A dynamic analysis of robot movements [30], [28] brings new aspects of influences on the board. The measurements are now not longer taken in a static position of the TCP to check if the programmed and real reached TCP positions are fitting.

The dynamic measurement of TCP movements is a much more complex task than static checks. Based on the sampling rate and resolution of the used measurement system, the movement speed of the robots TCP has to be adapted. Even the influences to the robot system are shifted to a more complex set of influences, basically to velocity-based forces like centrifugal force, coriolis force or mass inertia.

The following influences are only a small facet out of the wide compound of different dynamic error influences. They should give an overview and a feeling which influences can decrease the accuracy of an even high-accurate calibrated robots, down to an insufficient level for the application.

Swing over

A quite common task in todays robot applications is a movement along a 90-degree edge. This geometrical structure, constructed of two straight lines perpendicular to each other is used in nearly every robot program. The theoretical movement along this geometry should be an exact positioning of the TCP on this straight line in every point using a constant movement velocity.

In practical sense, this movement is quite easy structured, but for the robot impossible to move. Using a given constant velocity of the TCP and the prediction not to leave the teached path, the robot has to break this prediction due to several reasons. Assuming a theoretical case, that the robots static accuracy will be perfect, so that the path points are reached exactly during start.

Refering to figure 6, the robot will leave the programmed path directly after starting its movement in point P1. The acceleration of the robots TCP is causing in moments and forces, which take part that the robot arm will have deformations based on the limited stiffness of the robot arm parts.

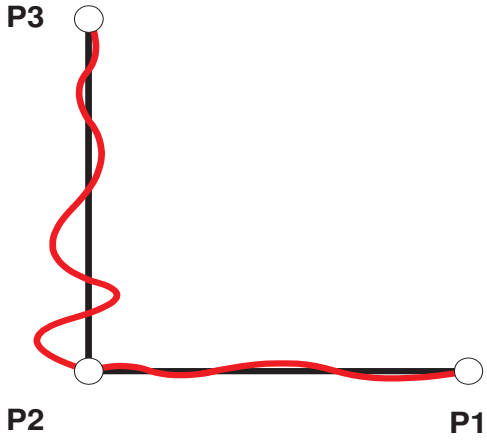

Figure 6: *Dynamic swing over*

Once accelerated, the movement of the TCP along the edge with constant speed is disabled by the possible maximum power of the joint motors, combined again with the flexibility of the arm parts. To change the movement direction of the robots TCP by 90 degree exactly at the edge in P2, a theoretical infinite acceleration of the TCP would be necessary for that.

This infinite acceleration can in reality not be fulfilled by the joint motors. Even in a theoretical case, if the motors would be provided with such power, the resulting moments and forces on the robot arm will be infinite and would cause positional deflections due to flexibilities.

The red movement, like shown in figure 6 is a common robot behavior and is on the one hand based on the arm flexibilities, on the other hand caused by the late reaction of the closed-loop position control of the robot. The swing over effect is increased by higher movement velocities and causing in todays applications still a needed compromise between path accuracy and movement velocity. The oscillation depicted in figure 6 can even be represented by a curve movement, cutting the program edge movement to decrease the needed moments of inertia.

Interpolation calculation

Another example effect of possible influences on the robot dynamic accuracy are the interpolation errors of the robot control [72], [41]. These can be focussed in a static and in a dynamic way of sight. In the static way, the interpolation of a geometrical structure - for example a semi-circle - is based on single fixed points, given to describe the geometry.

In case of a semi-circle, this points would be the points for start and end position of a curve movement followed by a middle position point to describe the radius of the curve. To interpolate this movement into many sampling positions between start and end point, the robot control will analyze the three given points, to calculate the shape of the semi-circle and all needed sampling points.

In a static way of sight, this is sufficient for moving the robot to each sampling position of the interpolation, with a single stop in every point. But taking consideration to a dynamic movement without stopping at this sampling positions and with even high-speed movements, it will be different. Besides of

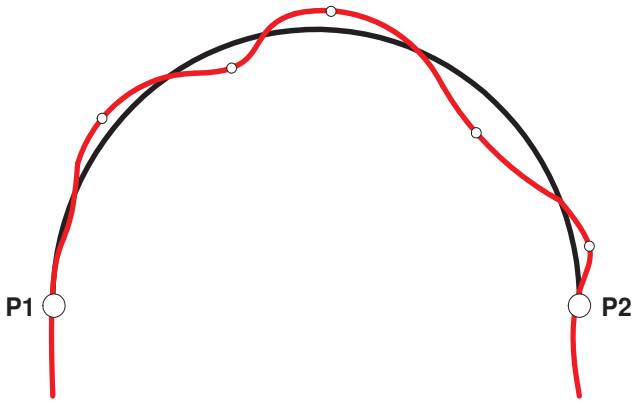

Figure 7: *Inaccurate IPO calculation*

swing-overs which have been mentioned before, the velocity-dependent influences on the robot arm are not completely taken into consideration by the robot control during interpolation. For this case, the interpolation result is not based on the real circumstances of the application, which leads to deviations in the movements.

3.5 Analysis of the robots accuracy

Qualitative predictions about robot accuracy are in nearly all cases strongly depending on the application. Robots in painting applications for example will have lower demands to the positional accuracy than robots in spot welding applications. To identify the needed accuracy parameters and its tolerances to guaranty a well working application, is in most of the cases combined with much effort in measurement and documentation.

In the next part of this chapter, an example analysis of a modern industrial robot is done, to show the practical meaning of acquiring and visualization of measurement data. The calculations are mostly based on the ISO 9283 to focus on the standard of determination of robot accuracy.

In later parts of this thesis (subchapter 5.1), an automated software is presented, which is generating a complete set of documentations and visualizations out of a measured set of robot test runs. The measurement results presented in the examples of this chapter are processed directly, using Microsoft Excel for numeric operations.

3.5.1 Robot base frame identification

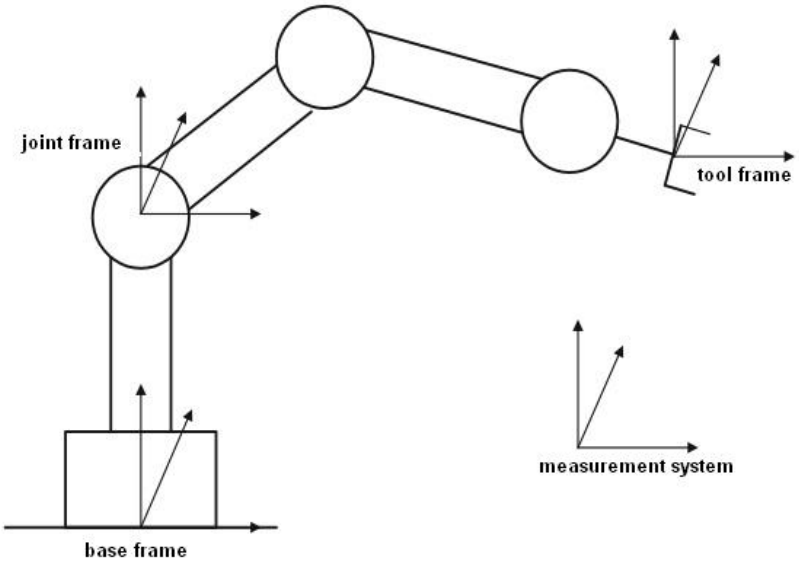

Figure 8: *Coordinate frames*

For a comparison of the measured TCP positions to the stored points in the robot program, it is needed to calculate the transformation between measurement system and robot. For this step, the position and orientation of the robot base frame has to be identified in the coordinate frame of the measurement system (see figure 8).

This identification can be done by measurement of several different TCP positions of the robot [38]. The measurements can be taken statically and the positions have to be spread into the robots working area, to verify an equal influence to the robots accuracy distributed in the used working space (figure 9).

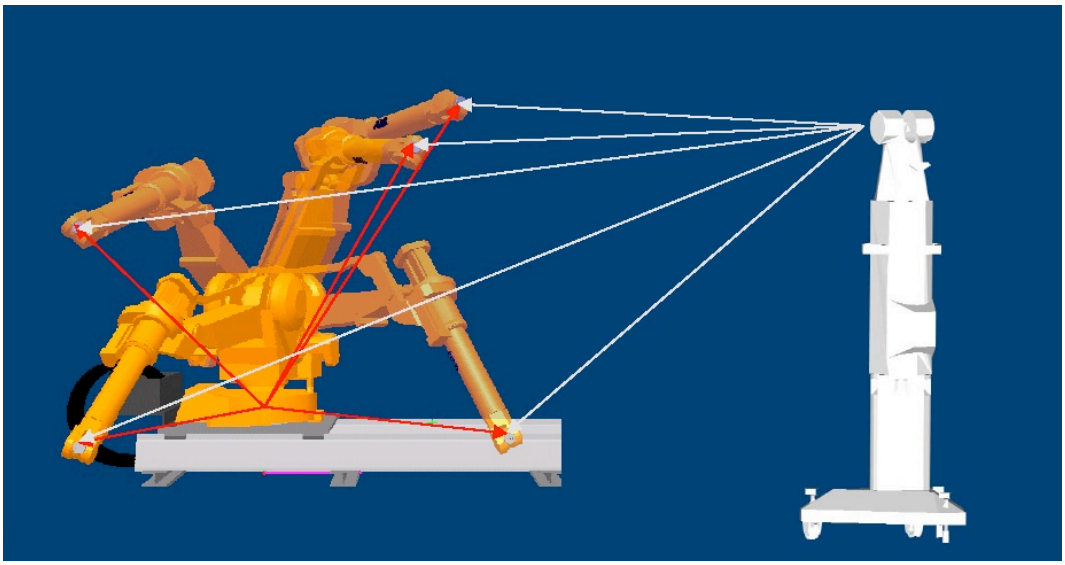

Figure 9: *Measurement of TCP points*

The measured point coordinates for the analyzation of the robots accuracy can be converted from the measurement frame representation into the robot base frame representation. This can be done by the measurement system directly during measurement. The only needed information is the 6-dimensional transformation from the measurement system frame to the robot base frame.

To calculate the transformation from the measurement system into the robot system, two different point sets are processed. The first is the set of measured points A, the second set are the programmed robots TCP positions B. The searched transformation is defined by ${}^{A}T_{B}$:

$$ {}^{A}\vec{p}_i = {}^{A}T_B \cdot {}^{B}\vec{p}_i \tag{24} $$

with

$$ {}^{A}T_B = \begin{pmatrix} \cos\gamma\cos\beta & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma & t_x \\ \sin\gamma\cos\beta & \sin\gamma\sin\beta\sin\alpha + \cos\alpha\cos\beta & \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha & t_y \\ -\sin\beta & \sin\alpha\cos\beta & \cos\beta\cos\alpha & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{25} $$

The matrix ${}^{A}T_B$ is depending on 6 unknown variables, three of them for cartesian shift (t_x, t_y, t_z), the other three for rotation (α, β, γ). Therefore:

$$ {}^{A}T_B = f(\alpha, \beta, \gamma, t_x, t_y, t_z) \tag{26} $$

The multiplication of ${}^{A}T_B$ with ${}^{B}\vec{p}_i$ results in an equation set of three different equations for each measured point:

$$ \begin{aligned} {}^{A}\vec{p}_{i,x} &= (\cos\gamma\cos\beta) \cdot {}^{B}\vec{p}_{i,x} + (\cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha) \cdot {}^{B}\vec{p}_{i,y} \\ &\quad + (\cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma) \cdot {}^{B}\vec{p}_{i,z} + t_x \end{aligned} \tag{27} $$

$$ \begin{aligned} {}^{A}\vec{p}_{i,y} &= (\sin\gamma\cos\beta) \cdot {}^{B}\vec{p}_{i,x} + (\sin\gamma\sin\beta\sin\alpha - \cos\alpha\cos\gamma) \cdot {}^{B}\vec{p}_{i,y} \\ &\quad + (\sin\gamma\sin\beta\cos\alpha - \sin\alpha\cos\gamma) \cdot {}^{B}\vec{p}_{i,z} + t_y \end{aligned} \tag{28} $$

$$ \begin{aligned} {}^{A}\vec{p}_{i,z} &= (-\sin\beta) \cdot {}^{B}\vec{p}_{i,x} + (\cos\beta\sin\alpha) \cdot {}^{B}\vec{p}_{i,y} \\ &\quad + (\cos\beta\cos\alpha) \cdot {}^{B}\vec{p}_{i,z} + t_z \end{aligned} \tag{29} $$

The $3n$ equations resulting from n measured positions, are building a huge set of non-linear equations. The basic idea of calculating the resulting transformation between measurement system and robot is to solve these equations.

A closed analytic solution for this problem is not possible due to the inaccuracies of the robot. Additionally, the measurement system has got an uncertainty for measurements [33], [34], which is more closely explained in chapter 6. Really exact, compared to the robots absolute accuracy, the measurement points can be assumed as fixed values.

Like already mentioned, the robot will not be able to move its TCP to the programmed point coordinates. Based on this fact, the used point set which is compared to the measurement points, is consisting of the desired TCP coordinates plus an unknown offset, caused by the robot inaccuracy.

Using a numeric method for linearization and solving of the equation set [55], [37], [48], the well-known Newton method is applied [12]. It is used to find the roots of the equations, which are changed to:

$$\begin{aligned} 0 &= (\cos\gamma\cos\beta) \cdot^B \vec{p}_{i,x} + (\cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha) \cdot^B \vec{p}_{i,y} \\ &\quad +(\cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma) \cdot^B \vec{p}_{i,z} + t_x -^A \vec{p}_{i,x} \end{aligned} \tag{30}$$

$$\begin{aligned} 0 &= (\sin\gamma\cos\beta) \cdot^B \vec{p}_{i,x} + (\sin\gamma\sin\beta\sin\alpha - \cos\alpha\cos\gamma) \cdot^B \vec{p}_{i,y} \\ &\quad +(\sin\gamma\sin\beta\cos\alpha - \sin\alpha\cos\gamma) \cdot^B \vec{p}_{i,z} + t_y -^A \vec{p}_{i,y} \end{aligned} \tag{31}$$

$$\begin{aligned} 0 &= (-\sin\beta) \cdot^B \vec{p}_{i,x} + (\cos\beta\sin\alpha) \cdot^B \vec{p}_{i,y} \\ &\quad +(\cos\beta\cos\alpha) \cdot^B \vec{p}_{i,z} + t_z -^A \vec{p}_{i,z} \end{aligned} \tag{32}$$

The solution of the equation set is represented by a 6-dimensional vector, consisting of the values $t_x, t_y, t_z, \alpha, \beta, \gamma$. Based on the over-determined character of the equation set and the number of different variables, the single functions can not be simply differentiated.

For construction of the differentiations for every equation in each direction, the Jacobian Matrix is used [67]. Built from all possible differentiations this matrix is:

$$J(\vec{x}) = \begin{pmatrix} \frac{\partial F_1}{\partial \alpha} & \frac{\partial F_1}{\partial \beta} & \frac{\partial F_1}{\partial \gamma} & \frac{\partial F_1}{\partial t_x} & \frac{\partial F_1}{\partial t_y} & \frac{\partial F_1}{\partial t_z} \\ \frac{\partial F_2}{\partial \alpha} & \frac{\partial F_2}{\partial \beta} & \frac{\partial F_2}{\partial \gamma} & \frac{\partial F_2}{\partial t_x} & \frac{\partial F_2}{\partial t_y} & \frac{\partial F_2}{\partial t_z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_{3n}}{\partial \alpha} & \frac{\partial F_{3n}}{\partial \beta} & \frac{\partial F_{3n}}{\partial \gamma} & \frac{\partial F_3 n}{\partial t_x} & \frac{\partial F_{3n}}{\partial t_y} & \frac{\partial F_{3n}}{\partial t_z} \end{pmatrix} \tag{33}$$

The F_n are representing the single functions of the $3n$ different equations, created by the n measured and programmed points. Assigned to the multi-dimensional case, the Newton formula changes to:

$$f(x) \approx f(x^{(n)}) + J(x^{(n)})(x - x^{(n)}) \tag{34}$$

The corresponding iteration description for $x^{(n+1)}$ is calculated by a substitution of $f(x)$ by zero and x by $x^{(n+1)}$:

$$0 = f(x^{(n)}) + J(x^{(n)})(x^{(n+1)} - x^{(n)}) \tag{35}$$

$$\Leftrightarrow \quad x^{(n+1)} = x^{(n)} - (J(x^{(n)}))^{-1} \cdot f(x^{(n)}) \tag{36}$$

For the construction of $x^{(n+1)}$ the Jacobian Matrix has to be inverted. Due to its non-symmetric structure, a simple inversion of the matrix is not possible. For this, the Pseudo Inverse of $J(x^{(n)})$ [69] can be used for calculation.

The presented algorithm for finding the transformation between measurement device and robot is based on the Newton method. Many other methods [49], [50], [51] for this calculation are known and can be applied to this problem, for example the least-squares fitting [12], which is faster in calculation. Because of usage of the transformation before measurement start and because of the fact, that it has only to be done one time, the standard algorithm is explained here.

3.5.2 Case study 1: static accuracy

This chapter is providing a more practical illustration, how the results of a static accuracy analysis looks like. As an example application, a high-accurate tripod robot was measured. The task of this robot is to work on a special plastic part to apply sealing mass into a small gap of the housing (see figure 10).

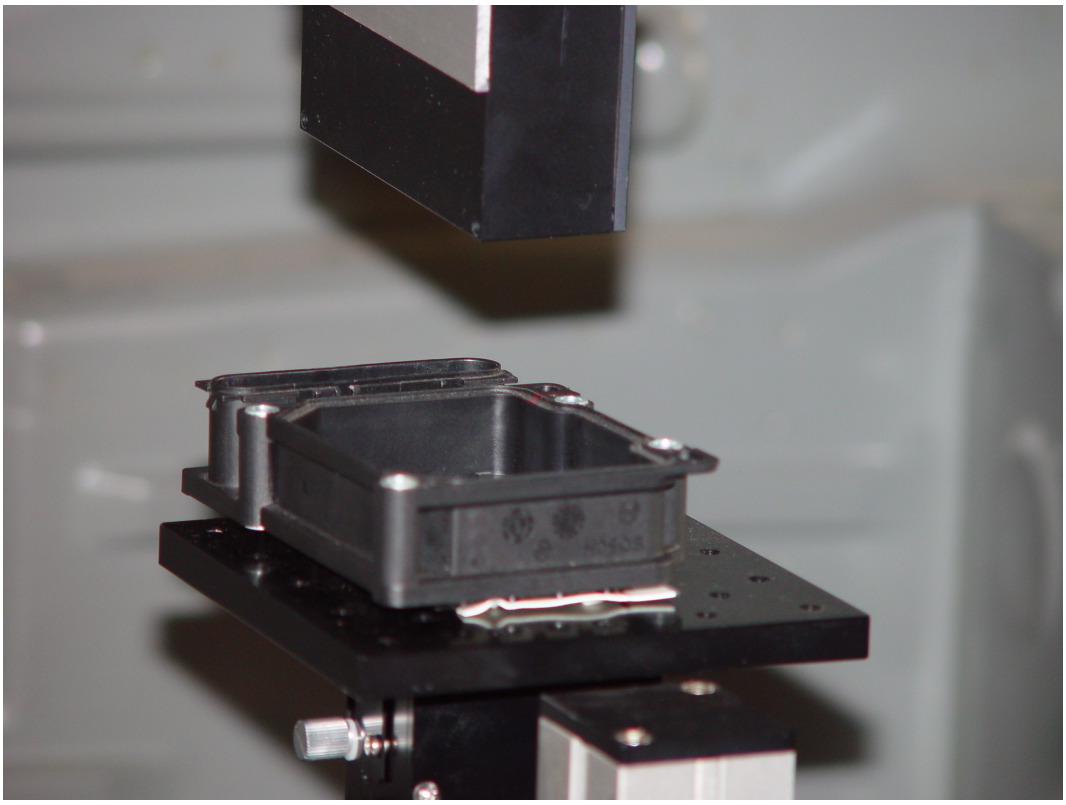

Figure 10: *Example application*

For a further improvement of the robots accuracy, it is using a laser-based external sensor, which is not taken into consideration for the accuracy test. The test is only done using the robot itself, while measuring its TCP movements.

Like described in chapter 3.5.1, at a first step, the robot base frame was identified. This enables a comparison between the measured and programmed TCP positions. At a next step, ten different TCP positions were measured in the application area of the robot. These points were programmed with different orientations, to emulate the real application behavior of the robot.

For all measurements, a non-contact measurement system consisting of a Leica laser tracker LTD800 was used. This system is capable to measure arbitrary 3D-positions in a measurement range of 15 meter in radius. A special corner cube mirror was used for the reflection of a laser beam and provided a centric reflection independent of the laser beam incidence angle. A very detailed description of this measurement system is given in subchapter 6.1.

Point	**x (mm)**	**y (mm)**	**z (mm)**
P_1	-200,00	200,00	200,00
P_2	-200,00	200,00	200,00
P_3	200,00	0	200,00
P_4	-200,00	0	300,00
P_5	200,00	0	300,00
P_6	200,00	200,00	300,00
P_7	-200,00	200,00	300,00
P_8	200,00	200,00	400,00
P_9	200,00	-200,00	400,00
P_{10}	-200,00	0	400,00

Table 1: *3D coordinates x,y and z of test points*

Regarding to table 1, the measured robot working range is a cube of length x = 400 mm, y = 400 mm and z = 200 mm. Compared to common sealing or welding robots in automotive industry, this is a small working range but related to the sealing task on the used application object it is adequate.

After measurement of the reference points for the robot base frame identification, the corresponding frame for transformation of the measurement results into robot base was calculated. After that, the 10 different programmed TCP positions were measured. This was done using 5 repetitions for each position, to calculate the center of gravity for each TCP position out of 5 measurements, see table 2:

	mean of measurements			**mean deviations**		
	x (mm)	**y (mm)**	**z (mm)**	**x (mm)**	**y (mm)**	**z (mm)**
P_1	-199,959	-200,063	200,003	0,04	-0,06	0,00
P_2	-199,646	199,705	200,046	0,35	-0,29	0,05
P_3	199,974	0,021	199,957	-0,03	0,02	-0.04
P_4	-200,064	-199,954	300,006	-0,06	0,05	0,01
P_5	199,897	-199,988	299,887	-0,10	0,01	-0,11
P_6	199,947	200,067	300,020	-0,05	0,07	0,02
P_7	-199,798	199,829	300,042	0,20	-0,17	0,04
P_8	199,890	200,160	400,035	-0,11	0,16	0,03
P_9	199,816	-199,924	399,897	-0,18	0,08	-0,10
P_{10}	-200,051	0,131	400,068	-0,05	0,13	0,07

Table 2: *Center-of-gravity of 5 iterations and deviations to programmed positions*

The fact, that using the ISO 9283 method for calculation the resulting accuracy is based on the mean values of measurements, a second calculation method was also applied, which is showing the real maximum error in positioning of the robots TCP, called "MAX" in table 3.

Because of the unique distribution of the positional deflections dependent to the robots TCP position, there is not a big difference between MAX and ISO calculation method in this case. Figure 11 is illustrating the analyzation results.

As one can see, the static positional deviation of the robot is situated between 0,05 mm and 0,5 mm.

	positional deviation		repeating deviation	
	ISO (mm)	**MAX (mm)**	**min (mm)**	**max (mm)**
P_1	0,08	0,08	0,008	0,012
P_2	0,46	0,47	0,003	0,010
P_3	0,05	0,06	0,007	0,018
P_4	0,08	0,08	0,004	0,009
P_5	0,15	0,16	0,003	0,011
P_6	0,09	0,10	0,003	0,015
P_7	0,27	0,28	0,003	0,014
P_8	0,20	0,20	0,009	0,015
P_9	0,22	0,23	0,003	0,014
P_{10}	0,16	0,16	0,006	0,016

Table 3: *Resulting deviations using ISO and MAX method*

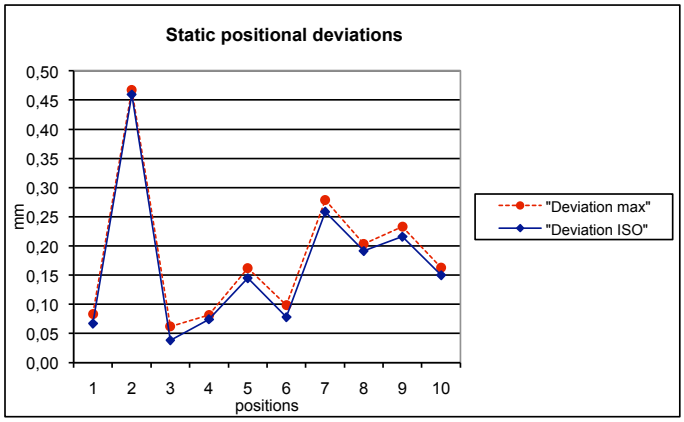

Figure 11: *Static positional accuracy diagram*

The repeating accuracy is calculated according to ISO 9283 and depicted in figure 11 using a separation between the best (deviation min) and the worst (deviation max) point out of 5 in every TCP position. The real repeating accuracy will be situated in the gap between those two values.

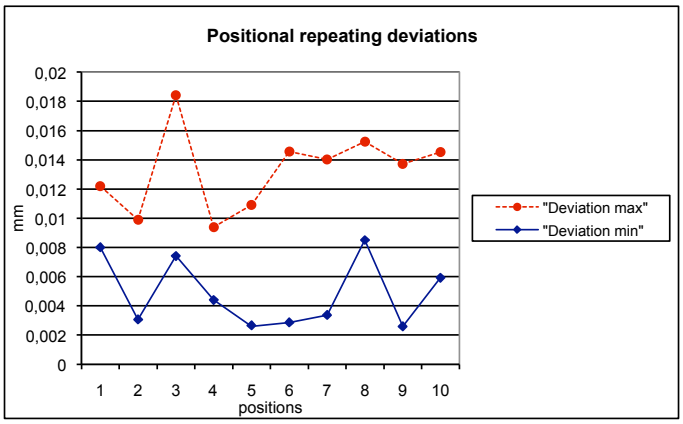

Figure 12: *Static repeating accuracy diagram*

3.5.3 Case study 2: straight line movement

The analysis of a straight line movement gives much information about the robots dynamic behavior [35]. Based on a simple geometry, special single dynamic influences can be separated and focused. The calculations are combined with higher effort in data processing, because the measurements are this time not taken statically, but with a high sampling rate during the movement.

The sampling rate of the dynamic measurements is dependent on the provided features of the measurement device and also influencing the meausurements, taking consideration to the sampling theorem. The Leica laser tracker, which was used in this thesis (see subchapter 6.1) is able to provide a sampling rate of 1000 measurements per second.

To process all data, an own developed Microsoft Excel sheet was developed, to optimize the numeric operations and to provide a visual depiction of the analyzation results. All algorithms for determination of the accuracies and deviations were implemented for a test case in Excel. Due to the structure of the used program, there is a lot of user interaction needed, to do one analyzation of a test run.

A further step of automation was reached in this thesis by the implementation of an automated processing of the measurement data (see subchapter 5.1). It comes with a minimum of user interaction and is not longer based on additional programs like Excel. For the calculations of the robots dynamic

behavior, the straight line movement defined by two three-dimensional coordinates $P_1(x_1, y_1, z_1)$ and $P_2(x_2, y_2, z_2)$ was implemented as a simple trajectory example.

Considering, that the coordinates of the robots TCP are defined 6-dimensional, the programmed orientations during this first simple movement were equal. Based on the fact, that the used measurement device was only able to measure 3-dimensional points, a practical check of the real orientations of the robots TCP was not possible.

Mathematically, the implemented straight line can be expressed as one 3-dimensional position $P_1(x_1, y_1, z_1)$ with an additionally given direction vector $\vec{R}$ (see figure 13).

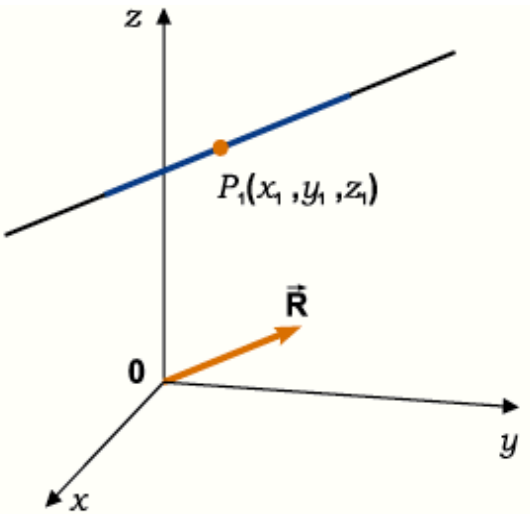

Figure 13: *Used mathematical definition of the straight line*

Using this, a straight line equation can be constructed, based on the component description method:

$$\frac{x - x_1}{l} = \frac{y - y_1}{m} = \frac{z - z_1}{n} \tag{37}$$

The absolute distance of every measurement point to the optimal straight line movement, is mostly defining the dynamic accuracy of the robot. To get this absolute distance, the perpendicular distance between the straight line and each measurement point is calculated:

$$d^2 = [(a-x_1)m-(b-y_1)l]^2+[(b-y_1)n-(c-z_1)m]^2+[(c-z_1)l-(a-x_1)n]^2 \tag{38}$$

if using a scaled direction vector $\vec{R}(l, m, n)$ with $|l| = 1$.

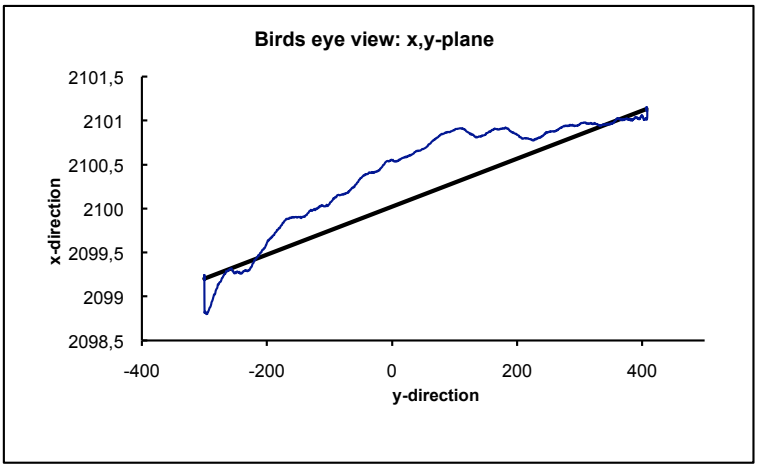

Figure 14: *Linear movement projected to 2D-plane*

The programmed movement path (black) and the real measured TCP-positions (blue) are depicted in figure 14. It is matched to a 2D-representation for an easy overview about the robots movement behavior. The differences d, calculated by equation 38 are presenting the real absolute deviations between TCP and straight line and are shown in figure 15:

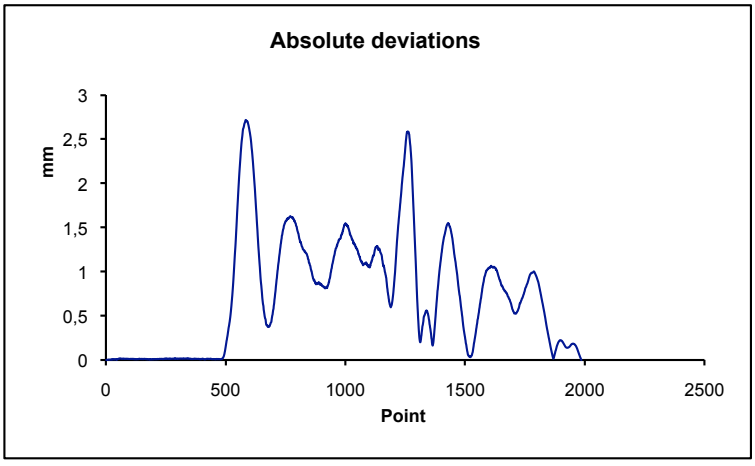

Figure 15: *Absolute differences from straight line*

The measurement data are showing one typical dynamic behavior of industrial robots. After start of the movement (lower left edge of figure 15), the flexibility effect mentioned in subchapter 3.4.2 is taking influence to the movement. This influence is causing a start-peak, effecting an absolute deviation value of more than 2.7 mm in this example.

The visualization of the 3D-measurements to enable proper analyzation by the user, is an important aspect of the documentation. Scaling and rotation of simple 3D-plots is in common sense influencing the subjective interpretation of the measurement data.

For providing an objective and easy-to-visualize way of data representation, a re-calculation of the measurement data into a special located and orientated coordinate frame was applied. The zero position of this coordinate frame is located in the starting point of the straight line movement. Also, it is rotated with the z-axis orientated equally to the straight line.

After re-calculation of all measured points into this coordinate frame, a "look-through" of the straight line movement is reached. In this, the start and end position of the movement are situated exactly at the origin of the new coordinate frame, only differing at their new z-value. Figure 16 shows the resulting diagram.

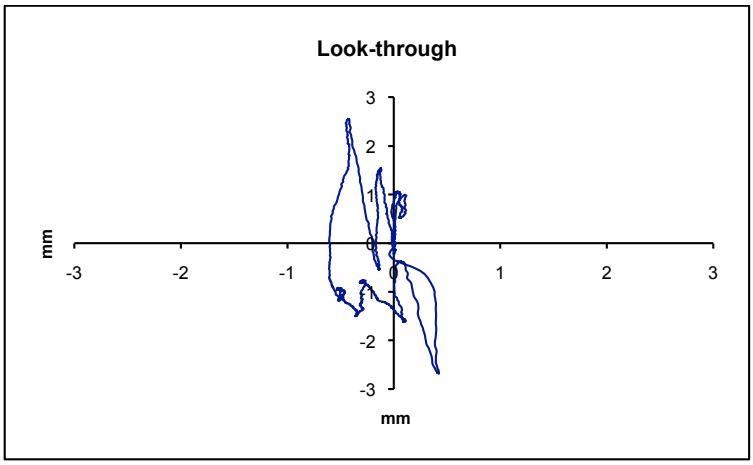

Figure 16: *Transformed measurement values into "look-through"-visualization with start and end position at diagram origin*

The look-through representation provides a possibility for a direct check of the measurement point distribution. A density distribution of the measured points is able to visualize parallel movement zones to the straight line. In case of a possible side shift of the TCP movement, the linear but shifted movement can be pointed out quickly by getting the point distributions.

Additionally, the vector length from the zero point to each single measurement point in the look-through diagram is equal to its absolute perpendicular deviation to the given straight line. Important to mention is, that the rotation of the diagram around its z-axis is not particular set in relation to the robot base.

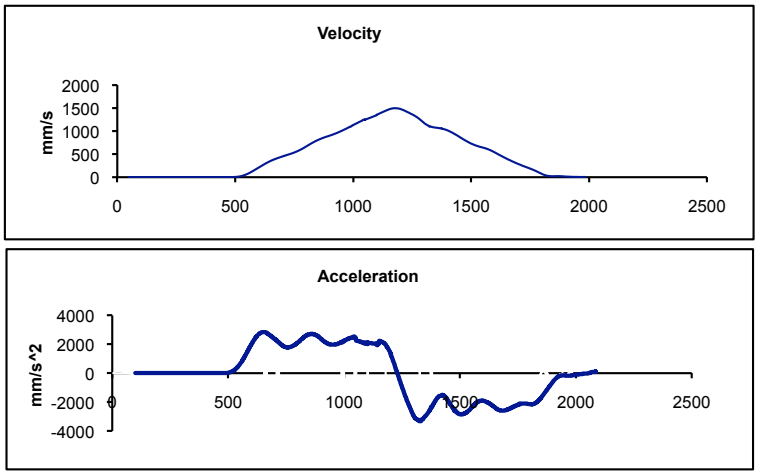

Figure 17: *Velocity and Acceleration diagrams calculated from the measured points during line movement*

Referring to figure 15, the positional deviation of the robots TCP from the straight line was approximately 2,7 mm at point 600 and following. Using the velocity and acceleration diagrams (figure 17) it can be seen, that in this zone, the acceleration of the robots TCP is at its maximum value and therefore the velocity is rising.

The interpretation of the relation between flexibility, acceleration and resulting positional deviation, can be used to provide the user with objective and easy to use information. Today this is used in an industrial application at an important german car manufacturer and besides this it is also automated to a autonomous system, presented in section 5.1.

4 Linear track

4.1 Introduction

Modern industrial robotics is affected by the rapid development of technologies in the field of automation. The increasing demands of customers to solve more and more complicated tasks using industrial robots, needs new solutions and strategies by the manufacturers. Terms like "flexible automation" express the new developments in the minds of the customers.

The workpieces, which an industrial robot is dealing with today are getting much more complicated than in the past years. There are many aspects to make a workpiece complicated, one of it is the size of the object. As mentioned, a significant part of the used robots all over the world are working in automotive industry. Thinking in manipulating parts of a normal car, like a front door for example, is not a big challenge today. But thinking in the sizes of a school bus, which has to be painted or welded by a robot is different.

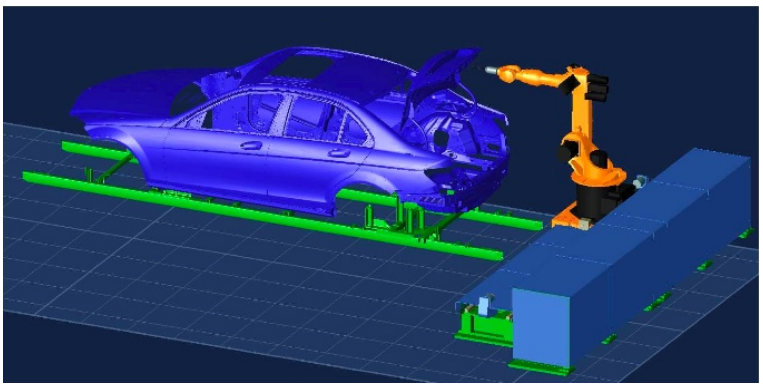

Figure 18: *Robot on linear track handling a large workpiece*

How to manage this size of workpieces is one of the modern tasks of robotics. Every robot has got a fix defined working range. Even if the working range of an industrial painting robot is quiet bigger than most of all other robots currently used all over the world, it will never be able to paint a school bus.

One solution of this problem is to combine several robots to work concurrently on one workpiece. The combination of the single working ranges of the robot to one big common working range allows to deal even with big objects. But the question is not only to realize a solution for this problem,

but also to be cost efficient. The combination of several robots working on one workobject is always a quite expensive solution. The question is, how to manage a big workpiece with a minimum number of robots? How to increase the robots working range with a simple, quick and reliable system?

One answer to this question, which is also a quite common solution in the flexible automation today is: Linear tracks! Linear tracks are special units which are able to move the robot itself in a linear movement. In most of the cases they are equipped with a motor driven sliding skid, moving on one or two iron tracks (see picture 19). Starting at movement distances of two

Figure 19: *Industrial robot on linear track*

meter - for example in automotive industry - the range of possible movement distances is nearly not limited. Linear tracks of even 50 meters are used in aircraft industry or for example in the construction of wind wheel wings for energy generation.

Using such a linear track, it is possible to adapt the working range of the robot to an optimal working position on the object. For this, the new possible working range of the robot is his own working range, elongated by the possible movement distance delivered by the linear track. For this, even very large objects can be handled and manipulated by a minimum number of robots. The linear movement of the robot on the track can be detected by the robot, due to movement sensors on the linear track, which are connected to the robot control. A highly accurate integration of the direction of the track profile in the robot motion planning system allows to calculate the exact position of the robot tool, even with movement of the linear track. How this is done and which different problems can occur, is explained in the following chapter.

4.2 Problems and tasks with linear tracks

4.2.1 Positional accuracy in y- and z-direction

One of the first problems during the process of combining an industrial robot with a linear track is the assembly of the linear track itself. Common linear tracks for industrial robots in automotive industry consist of metal rails, on which the robot is moving. Different varieties of combining robot and linear track appeared in the past. Beginning from hanging mountings, in which the robot is working overhead, ending with normal standing mountings, all different mounting angles between robot and track are possible.

One important aspect during the assembly of the track is to secure the straightness of the movement rails. A non-straight movement rail will cause positional deflections appearing on the robot TCP [31], [32]. The deviations from a theoretic perfect straight mounting of the rails can be expressed as non-linearities of the linear track. During the movement on a non-perfect rail, the robot will nick or gear in a non-linear way. Depending on the kind of non-linearity, the robot will react in a different way.

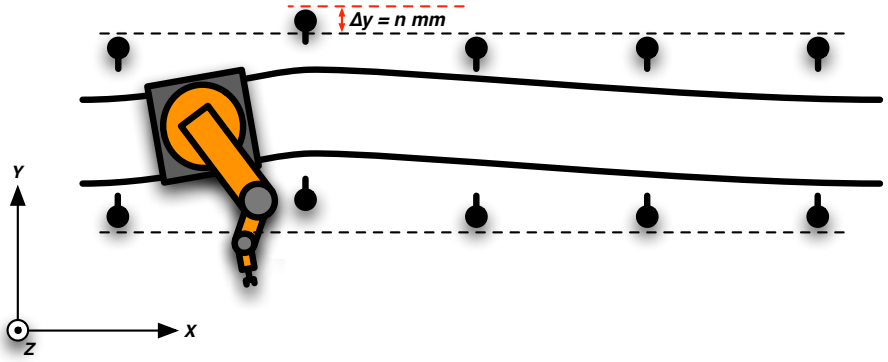

Figure 20: *Non-linearity caused by y-accuracy*

As shown in picture 20, one of the non-linearities is caused by a de-positioning of the fixing points of the track in y-direction. Caused by an assembly accuracy of several mm in y-direction, the application accuracy will be influenced by this value and never be able to reach accuracy states better than this. The error is a combination of the positional deflection caused by the side-shift effect and the rotational deflection caused by the gearing of the robot.

An comparable effect on the robots accuracy is caused by the height adjustment screws of the track. To adjust the linearity of the rails, it is neccessary to measure the height differences of the single screw positions. After that, an appropriate adjustment of the heigth can be done. Due to a limited exactness of the commonly used systems for measurement, the height profile of the rails includes non-linearities.

Likewise to the y-accuracy, an emerging side-shift and gearing cause in positional deflections on the robots TCP (see picture 21 for gearing effect).

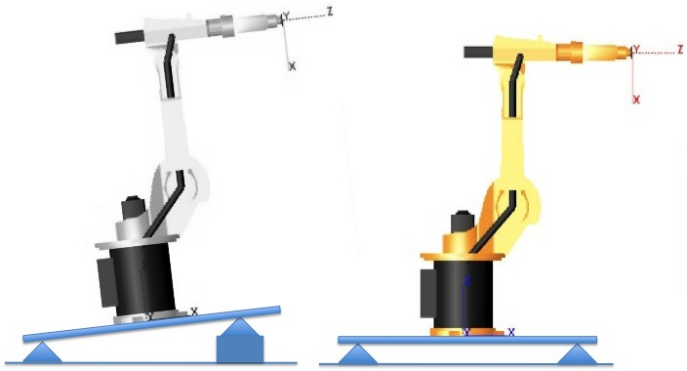

Figure 21: *Gearing effect due to height difference*

For this, an exact mounting of the linear tracks is directly improving the absolut accuracy of the robot application. Figure 22 shows a measurement of a robot moved on a linear track without movement of its joints. The measured TCP movement along y- and z-direction is only caused by the non-linearities of the linear track.

The common procedure of getting less influences caused by an inproper mounting of the track, is to improve the exactness of constructing and assembling. The emerging costs are proportional with the needed accuracy and in some cases the needed requirements are difficult to be fulfilled.

One solution for this problem is presented in subchapter 4.6 where the side-shifts and gearings are measured and expressed by a mathematical description of the track. Using this new method, it is possible to increase the systems accuracy without spending a lot of energy in calibrating the rails during assembly process.

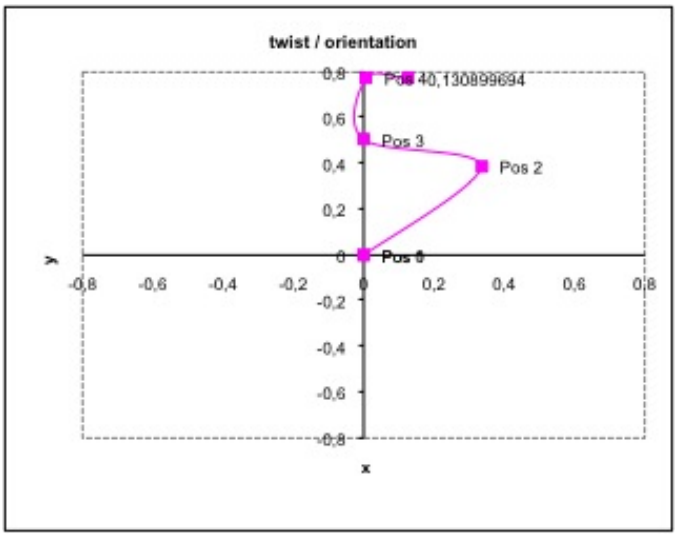

Figure 22: *TCP deflection in mm caused by linear track*

4.2.2 Alignment of the track in robot coordinate system

During movements of the robot along the linear track rails, the robot control calculates the 3D-position of the robots TCP. This 3D-position is depending on the robots arm configuration caused by the joint values and from the robots skid position on the linear track.

This position on the track has to be measured and used for the calculation of the relative position of the robot. In this section, several error influences, which are taking part of the main error of the application are explained more detailed.

During the setup process of an industrial robot, the alignment vector of the linear track in robot coordinate system has to be identified. This is normally done by an integrated routine of the robot control. The user needs to move the robots TCP to fix 3D-coordinates using different skid positions on the track (see figure 24).

With usage of 3 different skid positions with the same TCP positions, the robot control is able to calculate the direction of the linear track in the robots base coordinate system. For this a straight line least-square similar mathematical descriptions are used. One crucial effect of this methods is, that using the robot positions for the calculation of the track direction, the abso-

lute positional error of the robot will influence the measurements. Using the joint angle encoders for calculating the transformation into the TCP, there will always be an error, proportional to the positional accuracy of the robot.

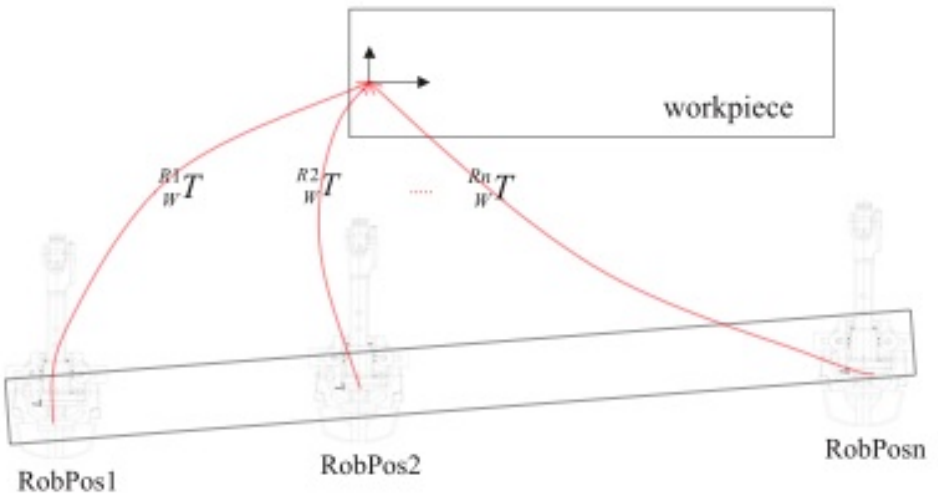

Figure 23: *Measuring in different positions on the skid*

4.2.3 Further effects

Additionally to the already mentioned effects, there are influences existing with lower consequence to the accuracy of the robot system. This effects are mostly seated in the system and can not easily be eliminated. The prior explained effects like an incorrect assembly of the track is normally based on human failing.

Gearfactor
Synchronised to the movement of the robot, the position of the skid on the track on which the robot is mounted, has to be detected. One of the most common method of getting the current position, is to use an incremental rotation counter. This counter is calibrated at a fixed zero position with an initial value. A movement of the motor shaft caused an increasing or decreasing of the rotation counter according to the skid movement.

The movement distance is calculated by the robot control, using the circumference of the gear wheel and the complete rotation angle (current angle position plus number of full rotations). The gear wheel circumference is represented by an multiplicative factor, so called "gear factor". After production of the linear track, the gear factor is set to its initial value corresponding to its theoretical size.

In practical application, this theoretical initial value is in most of the cases not accurate enough to guaranty a good positional accuracy of the linear track movement. It causes in a positional deflection of the robots TCP along the movement direction of the linear track, so in normal setup in one coordinate direction of the robots coordinate system. The positional error itself is because of its accumulating behavior depending of the movement distance, for that increasing with longer movements on the linear track.

Strategies for avoiding the positional deflection caused by an inaccurate gear factor, consist of an exact measurement of the movement distance along the linear track, followed by a calibration of the gear factor mulitplicator. See section 4.7 for more information.

The drag effect
To garanty an exact movement of the track, a theoretical reaction behaviour of the linear track motor has to be assumed. The feedback of a real electrical motor will always be later as the theoretical reaction, because of the response delay. For this, set point commands can only be processed in a defined reaction time.

The influence of this type of error is increasing with the dynamics of the robot. During fast movement changes it will be more significant than with slow movement changes. The resulting positional error of the robots TCP is caused by the difference between theoretical and pratical dynamic model of the robot. To get an adaption to the real model, extensive analysis has to be taken, to get a modified mathematical model for description of the dynamic effects during the drag effect.

The avoidance or minimisation of this influence is not explored in this thesis. It has to be tolerated or adapted by changing movement parameters in the robot control effecting the acceleration behavior of the track.

Gear loss
At constructions using a gear wheel to position the skid on the track, there is an error influence caused by the gap between gear wheel and bearing. The size of the gap assigns the resulting accuracy during seesaw movements. It is a compromise between durability and exactness. A very tight system using a reduced gap size will have more attrition than a loose system with bigger gap.

The resulting positional error will take effect during direction changes in the skid movement. The needed setting of the tightness of the system has to be adapted to the application needs.

Non-linearities
Caused by various influences of the single components of a linear track, a non-linear deflection behavior takes part on the positional error. Examples for this influences are the elasticity of the rail parts and fixings itself, or the non-linear interaction of force and current of the electric motor.

The influences are strongly dependent on the currently existing forces on the track skid, caused either by gravity, track movement of robot movement. For compensation or elimination of this influence, the mathematical model of the system robot and track together has to be changed. An experimental research of the response behavior during different test movements is possible, to get a prediction of the dynamic reactions.

4.3 Analysis of the geometrical structure of a linear track

4.3.1 Static analysis of the linear track

As mentioned in subchapter 4.2 the correctness of the position and construction of the track is directly dependent to the resulting accuracy of the robot system. For this reason, one important part of the assembly process of the linear track is to guaranty its correct positioning and structure. To certify this, measurements of the linear track structure on the one hand and the position of the linear track in space on the other hand have to be considered.

This measurements can be done in multiple and various ways, under usage of different kinds of measurement devices. one important feature of this devices is, that they are able to get at least 3D-coordinates in space. These 3D-coordinates can be used for proving the position of the track and for scanning the shape and the deformations of the track structure also. For generation of the results, which are presented in this thesis, a laser tracker measurement device was used. This device is able to measure 3D-coordinates using a laser beam in combination with a centric reflecting cube mirror. More information about the measurement device are given in chapter 6.

A major goal of this thesis is the minimisation of the error influences integrated in the application system robot with linear track. Necessary for this minimisation is a theoretical description of the linear track to develop a correction algorithm. This correction algorithm can be used for a compensation or reduction of error influences to increase the application accuracy.

To develop a strategy for description of the linear track, a logical proceeding structure leads along the pratical steps for analysation, theoretic model and finally compensation or reduction. In this case, the next steps of the analysis of the linear track are divided and structured under logical and practical aspects.

First of all, during assembling and adjusting of the track, the geometrical structure of the linear track itself, has to be measured and optimized. This is a common step during setup process of a linear track for industrial robot systems. Two different aspects in this case have to be considered, the positional and the geometrical aspect. Both aspects are influencing each other. Even if a linear track is highly accurate assembled and its structural deflections are negligible, its position in space possibly causes serious positional

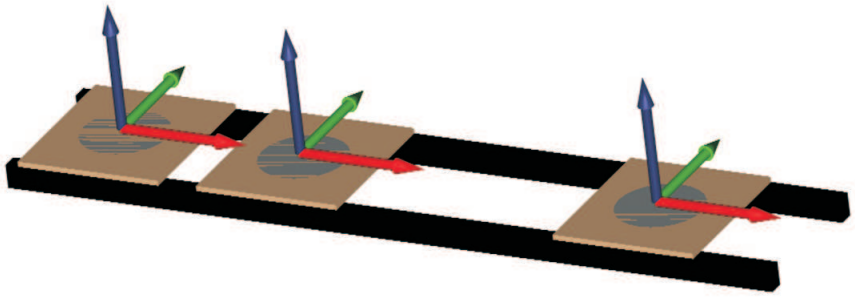

Figure 24: *Measuring in different positions on the skid*

errors, if the track is not aligned porperly into the robots coordinate system.

This effect is certainly particular grave, if the alignment of the track and robot is not done properly. Using an accurate alignment, even linear tracks which are mounted really angular in space can be used normally. The more important fact is to verify, that the track rails are really straight and parallel to each other. If they are not, the prior mentioned error effects take part.

The goal during the assembly of the track is to get as close as possible to the theoretical plannings and positions as possible. To facilitate this, a periodically repeated measurement of the track profile during assembly has to be done. In most of the cases, the work flow should be to start adjusting the rails at one side of the track and work sequentially to the other side of the track. After completion of the calibration and controlling work, a final check of the track has to be done to guaranty exactness.

After all, it has to be mentioned, that the measurement of the final assembled linear track is not fully representive for the application. A deformation of the track rails, caused by the robots weight and dynamics is not considered in this step of setting up the application. If the robot is moving along the linear track, mounted on the skid, dynamic forces depending on the robots current speed and movement will appear. This forces will cause in positional deflections of the robots TCP.

To reduce the influences of the robot weight, the linear track has to be measured while the robot is moving on it. This is impossible with common methods like measuring the track rails on different positions (see figure 24), because the most crucial measure positions are blocked by the robot itself.

For this reason, a measurement method has to be taken into consideration, where the linear track can be analyzed while the robot is mounted. The new solution for measurement of the track is based on a method using the robot itself as a rigid body for taking the measurements of the linear track. This method is described in the next section and is one of the contributions of this thesis.

The usage of an industrial robot, involved in a measurement process creates some problems concerning accuracy and repeatability. Subchapter 4.3.2 will take consideration of the key arguments for using the industrial robot as a rigid body.

4.3.2 Static analysis of linear track with robot

For a static analysis of all error influences contained in the application system consisting of robot and linear track, the measurements are taken under usage of the robot. The resulting positional deviations, which are important for the application accuracy can be determined at the robots TCP.

Getting the step to the measurement of the more complex system robot and linear track, instead of only the linear track itself gives more information and more possibilities for correction. But simultaneous with this, even more problems will affect the process of measurement. The static positional accuracy of the robot is limited and depending on the calibration of the robot (see subchapter 3.5.1).

This positional accuracy will influence the quality of the analysis of the linear track. On the one hand, the weight influence of the robot will be considered in the analysis of the track, on the other hand, the static positional accuracy of the robot is falsifying the results. To reduce the influence of the robots positional TCP deviations to the measurements, the robot is handled like a fixed rigid body for measurements.

One of the developed strategies of this thesis includes, that the robot is used for measurements but only in fixed positions. For this, the robot is moved to one arm configuration and measured at the first sampling point on the linear track. After that, it is moved only using the linear track to the next sampling point (see figure 26). The measurements are taken in equidistant points on the linear track. After one measurements series, the arm configuration of the robot can be changed and the process of measurement along the track can be repeated.

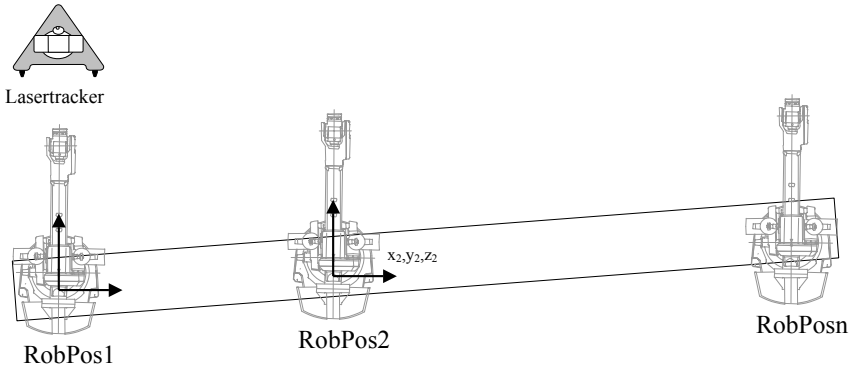

Figure 25: *Measurement of the robot TCP in series on the track*

Using this strategy, the forces caused by the arm configuration of the robot are taking influence to the linear track. Depending on one arm configuration it can be more or less force. After testing multiple arm configurations, a prediction of the flexible movements of the linear track can be done.

The unreliabilities in this setup are only the measurement uncertainty itself (see subchapter 6.2) and the repeating accuracy in positioning of the skid on the linear track.

4.3.3 Dynamic measurement of linear track with robot

As an extension of the static measurement of the track profile, a dynamic scan of the robots TCP movement enables more clearly conclusions about the track profile structure. In the breadboard, the robot got the reflector mounted on the TCP and moves with low speed on the skid along the linear track with a constant arm configuration. The reflector offset is integrated into new calculated TCP transformation data.

The resolution of the scan run is dependent on the sampling rate of the 3D-measurement device. The resulting scan is a quasi-continuos description of the track profile in one single arm configuration (figure 26), if the measurement rate is adequate. Now it is possible, to compare track scans with different arm configuration to each other, to see the influence of the robot arm position on the track profile. The comparison can only be done globally because an exact assignment of the single measurement points is based on a missing synchronisation difficult.
In the continuos scan, the dynamic influences of the linear skid movement

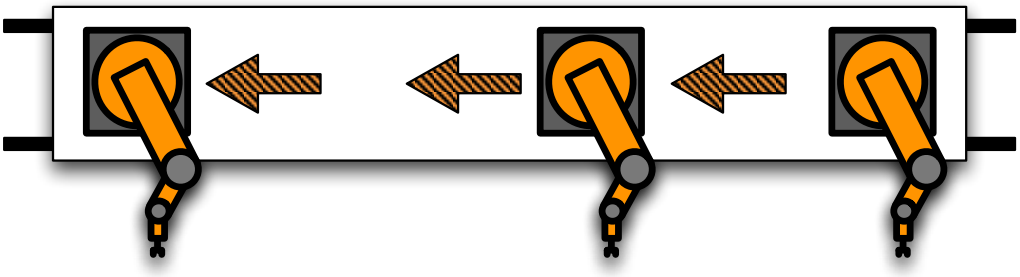

Figure 26: *Continuos scan of the linear track*

and the dynamic of the track rails during movement are implied. Even the couplings with the robot itself - apart from the movement dynamics - are integrated in the analysis result.

For a more detailed description, different robot movements can be fulfilled during skid movement, to get even the additional dynamics caused by the robot movements. A description of the dynamics and a mathematical description of the error influences caused by the robot during its movement are due to the non-linear couplings another complex task and are discussed in [69].

4.4 Measurement of the track profile

As mentioned before, the measurements of the track can be taken using the robot like a rigid body. For improvement of the applications accuracy, in this thesis a new method was developed for measurement and theoretical description of the linear track.

At a first step, the track is measured using the robot as a rigid body. The whole process can be divided into several different steps, which are constitutive on each other. If the steps are not carried out in the predefined order, an error propagation would decrease or even destroy the resulting accuracy win.

First, the measurement process is described, followed by the explanation of the theoretical description, based on the acquired measurement results.

TCP-calibration
The measurements for determination of the robots base frame are taken, using a measurement tool e.g. a cubic mirror. This mirror has to be mounted on the robot end effector. For the calculation of the reflector position at the robots flange, the so called "tool transformation" has to be calculated. This can be done by usage of the robot controller integrated function for identification of the tool frame, which is available in every common industrial robot type.

By this routine, three different arm configurations of the robot with an equal position of the tool center point have to be moved by the user and stored into the robot control. In practical application without access to measurement systems, this three positions are reached by moving peak-to-peak. The only check for an accurate movement is in that case the robot user.

Enhanced by a highly-accurate 3D-measurement system, the setup of the tool frame was done in this thesis using a special method. The fixed reference point, to which the robot has to be moved to, using three different arm configurations, was set as a virtual point. Once taken one measurement, the following robot tool positions after re-configuring of the arm status were measured and the difference vector was calculated:

$$\vec{p}_{fix} - \vec{p}_{rob} + \vec{u}_{meas} = \vec{r} \tag{39}$$

$$\Leftrightarrow \begin{pmatrix} p_{fix,x} \\ p_{fix,y} \\ p_{fix,z} \end{pmatrix} - \begin{pmatrix} p_{rob,x} \\ p_{rob,y} \\ p_{rob,z} \end{pmatrix} + \begin{pmatrix} u_{meas,x} \\ u_{meas,y} \\ u_{meas,z} \end{pmatrix} = \vec{r} \tag{40}$$

where $\vec{u}_{meas}$ is the uncertainty between real robot TCP position and the measured TCP position. Now following the goal to decrease the resulting distance vector $\vec{r}$ to the fixed position to zero, the robot position can be saved in the robot control if the break condition $|\vec{r}| < \epsilon$ is reached, with:

$$|\vec{r}| = \sqrt{(p_{fix,x} - p_{rob,x} + u_{meas,x})^2 + (p_{fix,y} - p_{rob,y} + u_{meas,y})^2 + (p_{fix,z} - p_{rob,z} + u_{meas,z})^2} \tag{41}$$

After saving the three different positions in the robot control, the integrated

algorithm calculates the transformation from flange to the tool center point. This transformation is stored into the robot control and is used for future position calculations.

To check, if the resulting transformation is correct, a rotational movement around the tool center point can be measured. With common industrial robot models, such movement functions are integrated in the standard set of movement commands. The measured deviation during this rotational movement is a resulting error component for further calculations.

A more accurate, but in most of the cases difficult to realize tool center point calculation, is a direct measurement of the tool center position. Beginning from reference points at the robot flange, the application points have to be measured exactly. In many applications this is not possible because of a non-touchable tool center point, e.g. painting applications.

Gear factor calibration

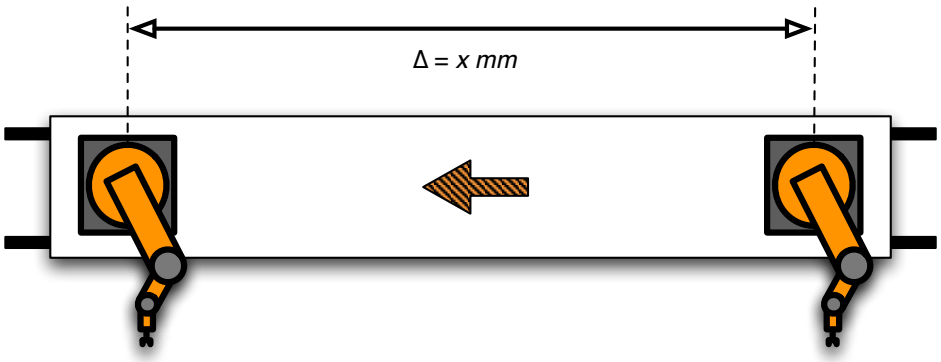

Figure 27: *Adjustment of the gear factor*

To get accurate movement steps during the measurements on the linear track, the gearfactor multiplier has to be set. As mentioned in subchapter 4.2.3 the gear factor is influencing the accuracy between commanded distance to real movement distance of the linear track skid (figure 27). If the gear factor is not calibrated exactly, the measurement values are not representative because they are falsified by the deviation offset between real and correct gear factor setting.

Beginning from a starting position at one side of the track, the robot is

moved to the other side of the linear track. In the start and the end position, the robots end effector, in this case the ball reflector, is measured. The robots arm configuration during the movement has to be constant to guaranty a measured movement distanced only caused by the skid shift.

Proportional with the moved distance of the track skid, the error of the calculated result will be increasing. After measurement of the moved distance, the new gear factor multiplier can be calculated as follows:

$$Gear_{new} = \frac{d_{meas}}{d_{rob}} \cdot Gear_{rob} \tag{42}$$

where d_{meas} is the measured distance and d_{rob} as commanded distance. The new factor is stored into the robot control for further measurements and can be tested in a second measurement run. If the resulting deviation is not the expected one, the calibration run can be done iteratively, until the residuum is reaching the breaking condition:

$$|d_{meas} - d_{rob}| = |r_{gear}| < \epsilon \tag{43}$$

The residual value after the iterative calibration is part of the uncertainty of the system.

Robot base coordinate frame

The robot is moved to ten different positions, spreat equally in its working range. In every position, the robot TCP - now the reflector of the measurement system - is measured. The programmed movement positions of the robot in cartesian representation, which are saved in the robot control have to be also saved allocated to the measurement values.

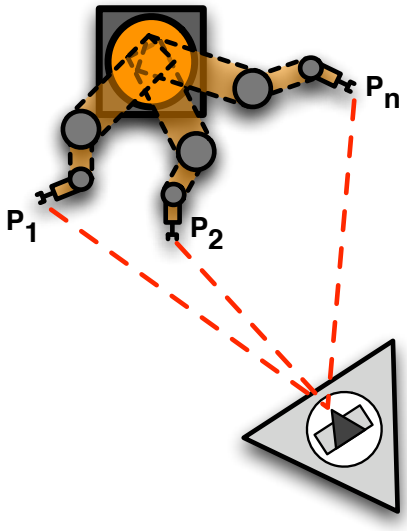

Figure 28: Identification of robot base coordinate frame in one position

After measurement n positions of the robot (the number of measurements is proportional to the resulting accuracy of the robots base, see [8]) a non-linear least-squares calculation is apllied to determine the robot base frame. Instead of a 3D-description of single measurement positions on the linear track itself, it is now possible to describe the linear track with 6D-represantation in one sampling point with concurrent consideration of the robots influence.

This influence is due to the multiple movement positions determined as average influence value. This is representive, because this influence will even take part in the application accuracy as an averaged value during real application. If the robot is working in a fixed known sub-space of its real working

range, the measurement positions for identification of the robot base should be shifted into the sub-range. This brings the theoretical model for correction of the linear track error influence closer to the real situation.

Shift of the robot base

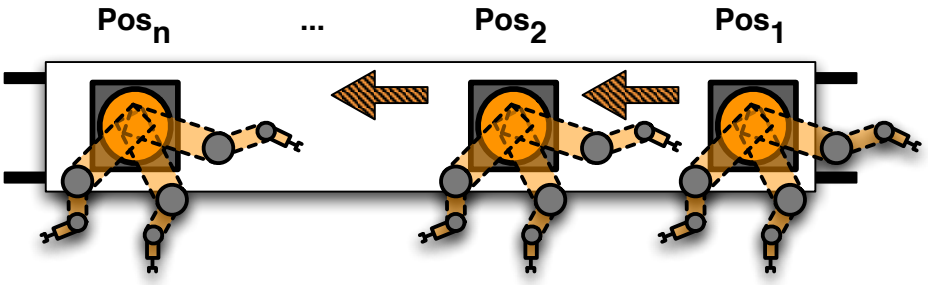

Figure 29: *Periodical identification of robot base in equidistant sampling points*

After measurement of the robots coordinate system in one sampling position, the procedure is repeated in equidistant sampling positions on the track (see figure 29). In every sampling position, the robot is measured in the same TCP position to exclude the absolute positional accuracy of the robots TCP. Using this strategy, the robot can be declared as a nearly rigid body because he is only used in its repeatability.

The constant distance between the sampling positions is depending on the accuracy demands. For determination of the needed distance between the sampling position, a frequency analysis of the track can be used to identify the amplitudes of single error parts in the track deformations. For a manual correction method (section 4.7) a rough raster of the sampling positions can be done ($<$10 for one track).

4.5 Frequency analysis of the linear track

A movement at a very low speed with a constant joint configuration of the robot arm is used for a high-resolution measurement scan. The movement is done in a straight line, only by the track itself to get only the structure of the track analyzed without influence of the robots movement. Also the movement speed V has not only to be low, but even constant. The corresponding scan mode of the measurement device is even switched to high resolution and discrete in time with a sampling interval of Δt.

For the measurement setup, one coordinate axis of the robot has to be in the movement direction of the linear track. If this is verified, the sum of the 3D-coordinates, measured during the skid movement represents the commanded function and the non-linear twist motion which is of interest. The resulting measured ramp function s_k consists of the sum of the single coordinates x_k, y_k and z_k.

After elimination of the commanded ramp function out of the sum of the coordinate values, the Fourier transformation [57], [58] of the h_k's contains the spectrum of the non-linearities of the track:

$$s_k = k \cdot \Delta, \quad k = 0, 1, ..., N-1 \tag{44}$$

and

$$\Delta = V \cdot dt \tag{45}$$

follows

$$h_k = x_k + y_k + z_k - s_k \tag{46}$$

where the x_k, y_k and z_k are the 3D scan of the TCP (see figure 30).

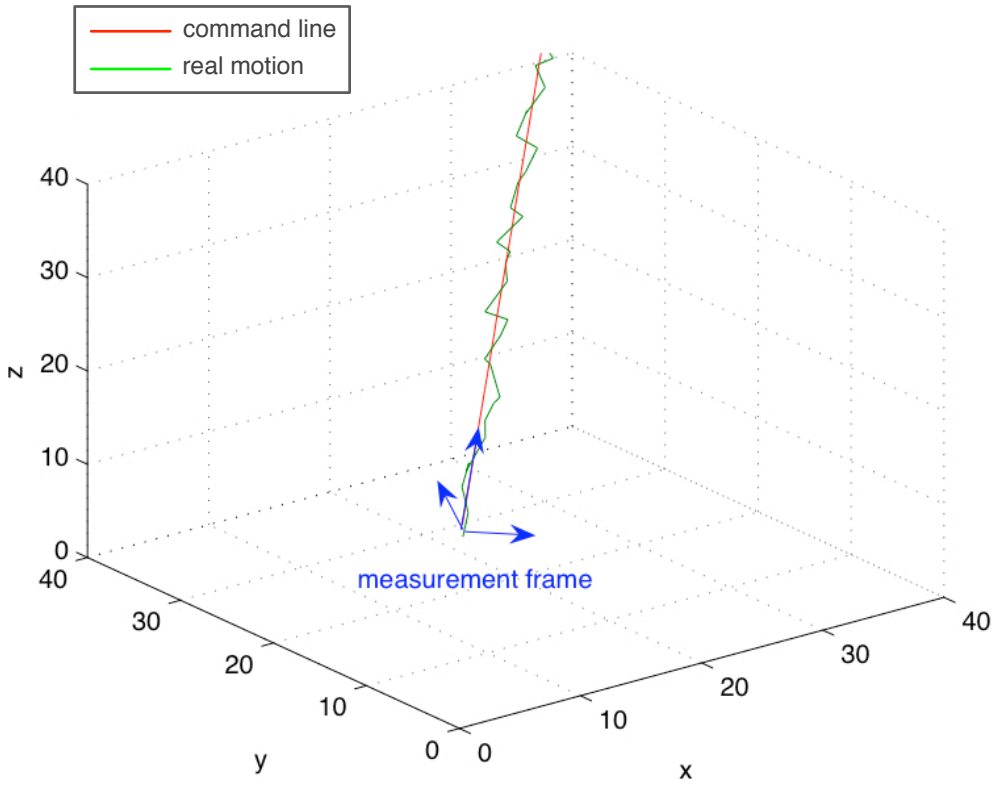

Figure 30: *Command and real motion*

The discrete Fourier transformation (DFT) [62] becomes

$$H(f_n) = X(f_n) + Y(f_n) + Z(f_n) - S(f_n) \tag{47}$$

where $f(n)$ is:

$$f_n = \frac{n}{N\Delta}; n = -\frac{N}{2}, ..., \frac{N}{2} \tag{48}$$

follows that

$$\begin{aligned} H(f_n) &= \int_{-\infty}^{\infty} h(s) e^{-j2\pi f_n s} ds \\ &\approx \sum_{k=0}^{N-1} h_k e^{-j2\pi f_n s_k} \Delta \\ &= \Delta \sum_{k=0}^{N-1} h_k e^{-j2\pi kn/N} \end{aligned} \tag{49}$$

From eq. 48 it is obvious, that the frequency is a spatial frequency of the unit mm^{-1}.

If the needed detection uncertainties for example is at 0.5 mm, so that all errors to detect are greater than this value, the resulting frequency of the sampling points can be identified using figure 31.

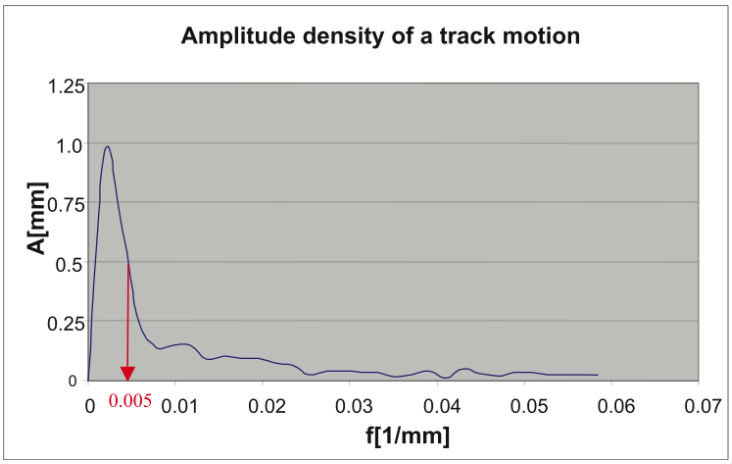

Figure 31: *Example of a track spectrum*

Respecting the sampling theorem [16]

$$T_s(m) \leq \frac{T_{min}(m)}{2} \tag{50}$$

it is essential to identify the robots coordinate frame each 100mm ($= 1/(2mm \cdot 0.005mm^{-1})$)

Frequency analysis measurements

For an example of a 3-dim frequency analysis, different robot movements were measured. The used robot was an ABB IRB 2400, mounted on a 1800 mm linear track (7^{th} axis). It is a middle-size robot for industrial applications like sealing for example.

The robot was programmed to move three different paths: one linear movement without using the 7^th axis, one linear movement with using the 7^{th} axis and one run with only $7th$ axis moving. The robot was moved with two different velocities, 200 mm/s and 600 mm/s.

The used measurement system for the needed measurement tasks like continuous scan of one track motion, determining the particular track coordinate systems or identifying of the robots accuracy is represented by a Leica Laser Tracker LTD800 (figure 32). The measurement device is presented more closely in subchapter 6.1.

Figure 32: *Leica LTD800 Laser Tracker*

For using an adequate window function for the time-limited DFT of the measured values, a comparison of some common used window functions (Kaiser, Hanning, Blackman-Harris, Gauss, Tukey)[16] was taken. The origin of the comparison was the movement of the robot with using the 7^{th} axis.

In figures 33 and 34 you can see the DFT of the robot movements with an used tukey window function and without any window function. Figure 34 is zoomed to the area after the main peak for better overview.

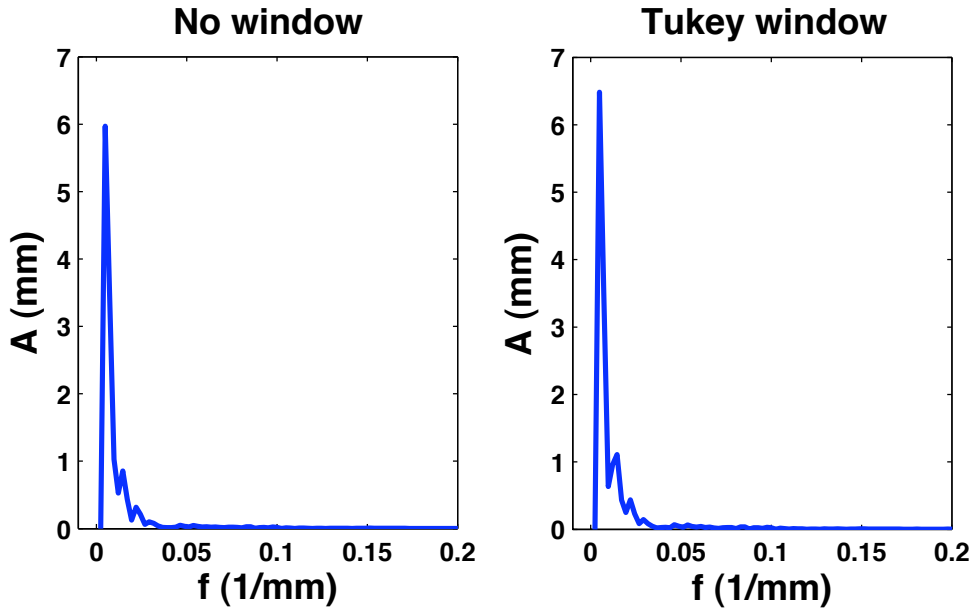

Figure 33: *Comparison of window functions*

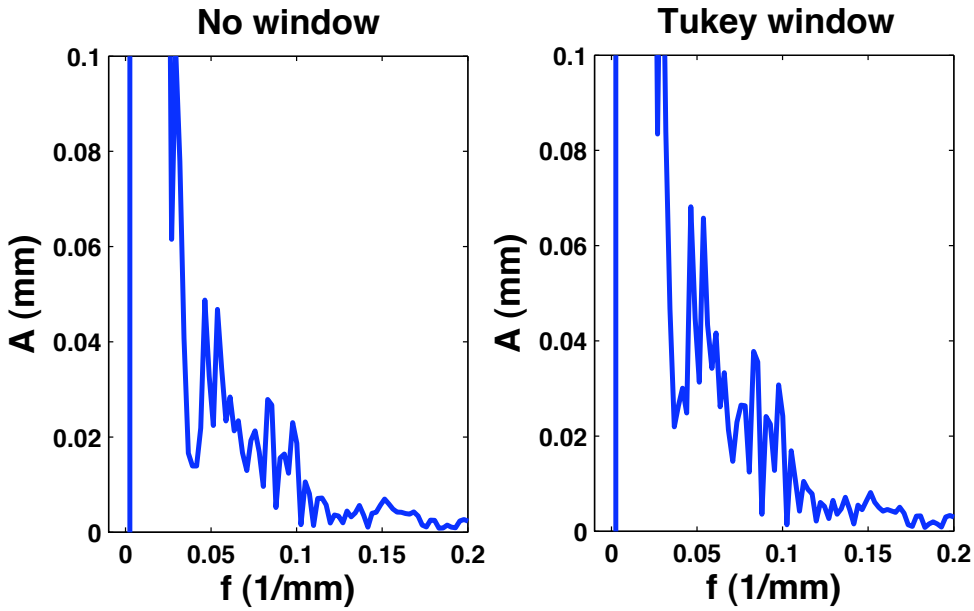

Figure 34: *Zoomed depicting*

The Tukey window has got a very good resolution of the first side peak at 0.025/mm. Even for its good side coil decay, the Tukey function was chosen as used window function because of its minimum leakage effect.

Robot linear movement

At the robot linear movements (figure 35) you can see, that there are more than one frequencies in the movement. The frequencies have different spectral lines, depending on the movement velocity of the track skid.

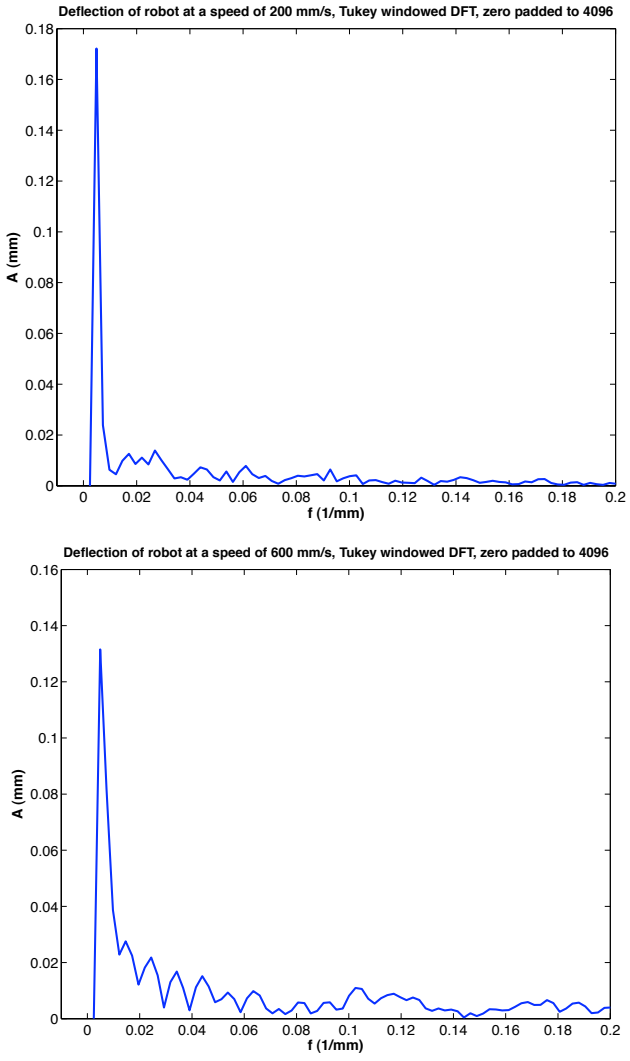

Figure 35: *DFT of robot linear movement at 200 mm/s and 600 mm/s*

The main amplitude of the robot movement is between 0.13 and 0.18 mm, depending on the movement speed, which can be seen on the deflection diagrams in figure 36. This deflection is the absolute value of the difference between programmed straight line movement and the measured trajectory of the robot.

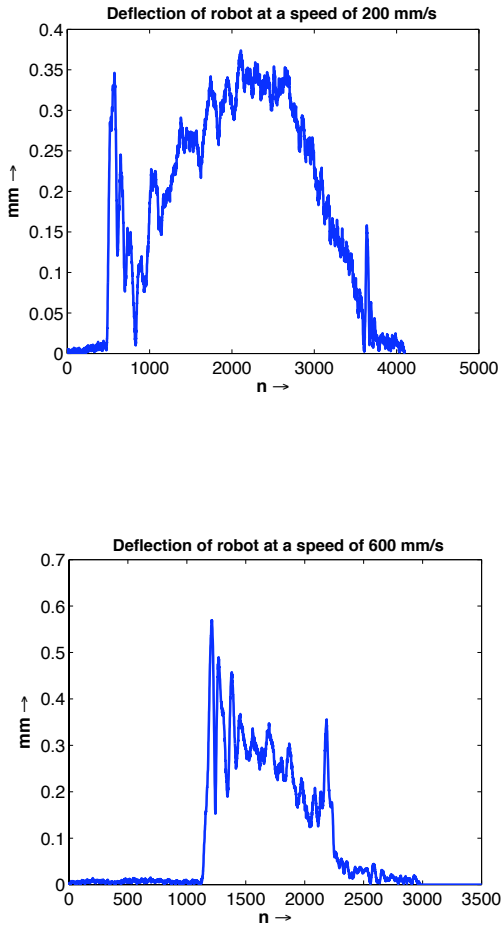

Figure 36: *Deflection to the programmed path at 200 mm/s and 600 mm/s*

Robot with linear track

If the robot is moving including the 7^{th} axis, a higher amplitude of the main peak in the DFT (figure 37) is resulting. Therefore, it has the greatest deflection from the programmed path (figure 38).This high deflection is caused by the combination of robot movement start and linear track movement start. Due to moments of inertia during synchronous movements of the linear track and robot, the demands on the control task are rapidly increased.

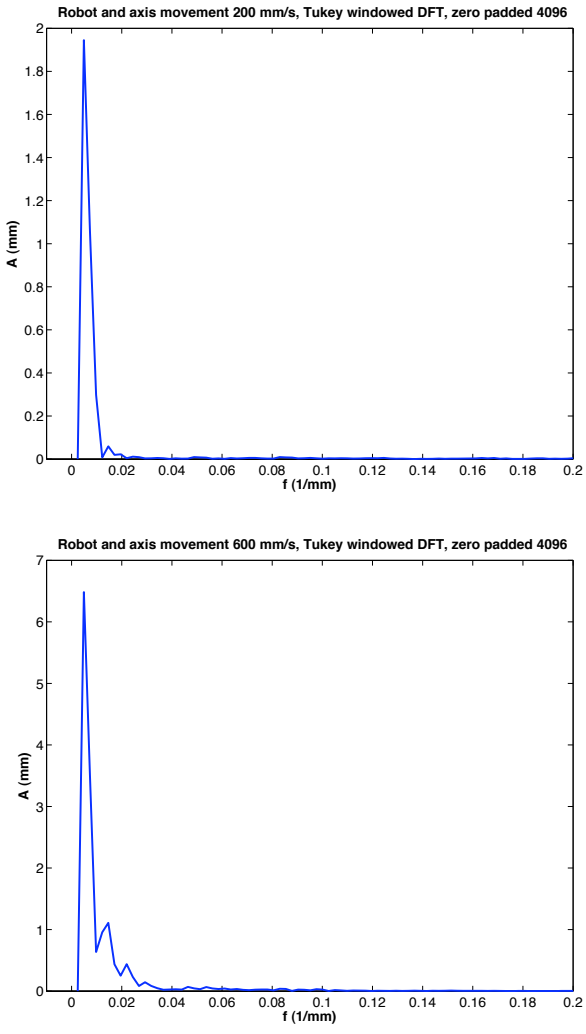

Figure 37: *DFT of robot with 7^{th} axis moving*

The last part of the deflection diagram for the movement with 200 mm/s is cut away due to a measurement stop, but the main part of the signal can be identified.

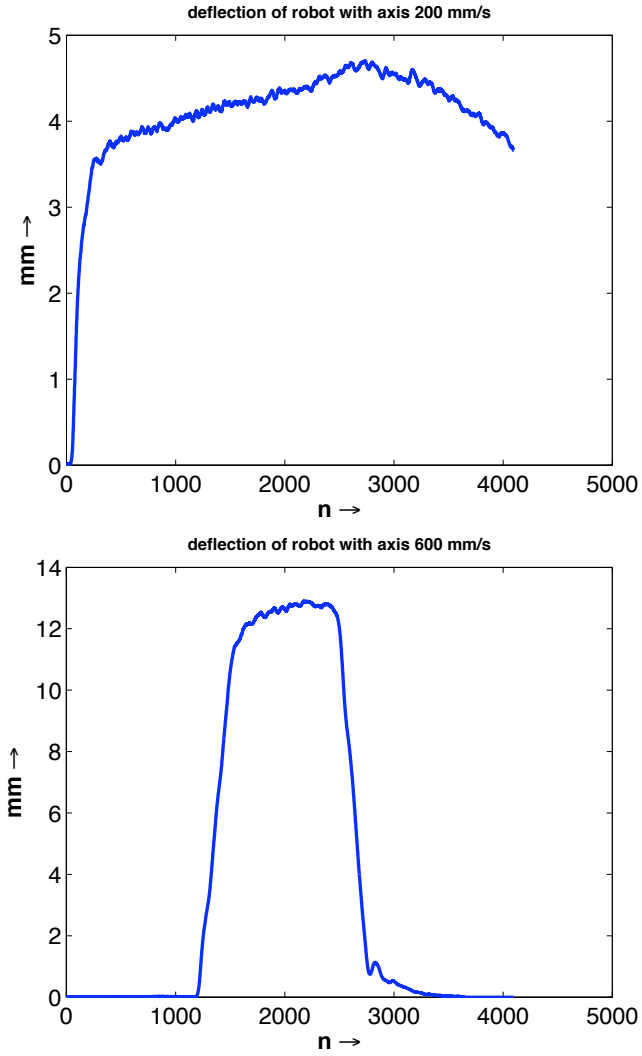

Figure 38: *Deflection of robot with 7^{th} axis moving*

Linear track movement

The linear track movement was done with a velocity of 250mm/s, figure 39 shows the discrete Fourier transformation of this movement. The deflection of the linear (figure 40) track is not only depending on the moments of inertia, but even from the geometrical configuration of the track (straightness of the rails and their horizontal alignment and parallelism).

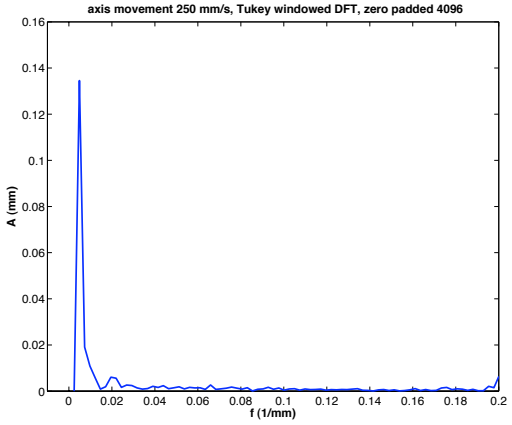

Figure 39: DFT of the linear track movement

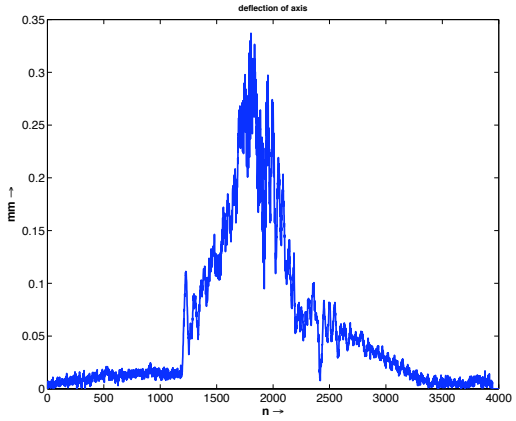

Figure 40: *Deflection of the linear track movement*

After all, the deflection of the linear track is like the deflection of the robot itself if slowly moved (250 mm/s). Only if both together, the robot and the track are moving the deflection is increasing.

Measurement Results

The Fast Fourier-Transformation (FFT) was applied in different tests, with different window functions for the time-limited spectral analysis, which were compared using measurement results of an industrial robot movement. At last, different DFT results of robot and linear track movements were presented.

The analysing of robot movements using Fast Fourier Transformation is an adequate method for predictions of the dynamic behaviour of the robot. Different oscillations of the robot arm can be tested with varying velocities or masses.

For the linear track, there can be made statements about the linearity and parallelism of the rails. The sampling rate for a compensation of the inaccuracies of the linear track can be determined, due to the maximal error for correction.

4.6 Mathematical description

Measurement of the linear track is the first step of the correction process. In the next step it is essential, to use the acquired measurement data for a mathematical description. To calculate the needed coordinate frames in the sampling positions on the track, represented by the robot base frame shifted in equidistant steps, two different sets of measurement data (figure 41) have to be fitted together [55], [52], [53], to find the final transformation between their two coordinate systems.

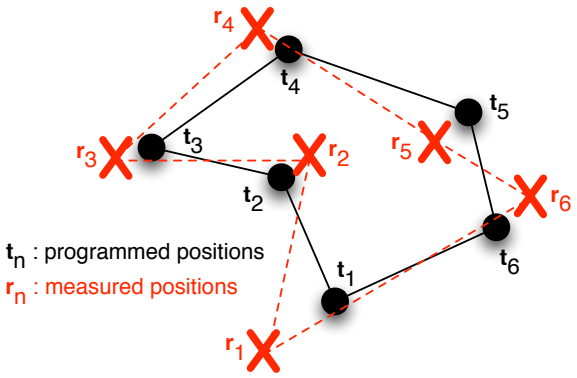

Figure 41: *Two sets of measurement positions*

The first data set are the values taken from the 3D laser tracker device. In case of n different positions, to which the robot was moved to during base frame identification, this are the n coordinates of the robots tool center point represented in the tracker system:

$$\vec{P}_{tracker} = \begin{pmatrix} \vec{p}_{1,t} \\ \vec{p}_{2,t} \\ \vdots \\ \vec{p}_{n,t} \end{pmatrix} \tag{51}$$

For determination of the 3D position of the robots tool center point, the robot control is calculating the forward transformation using the measurement values from each of its angle encoders. This gives also n measurement values:

$$\vec{Q}_{robot} = \begin{pmatrix} \vec{p}_{1,r} \\ \vec{p}_{2,r} \\ \vdots \\ \vec{p}_{n,r} \end{pmatrix} \tag{52}$$

These two data sets describe the same positions in space in two different coordinate systems including a deviation error. This error is caused on the one side by the trackers measurement uncertainty, on the other side by the robots inaccuracy. For this reason, these sets are only theoretical fitting to each other, practical there is a disturbance included.

For getting the comparison of the coordinate systems in the sampling points of the linear track, which are used to describe the non-linearities of the track, the transformation between tracker and robot in each sampling position is needed. To find this transformation, an optimal fitting of the two sets has to be calculated.

In the figure 42, the appropriate transformation graph is shown:

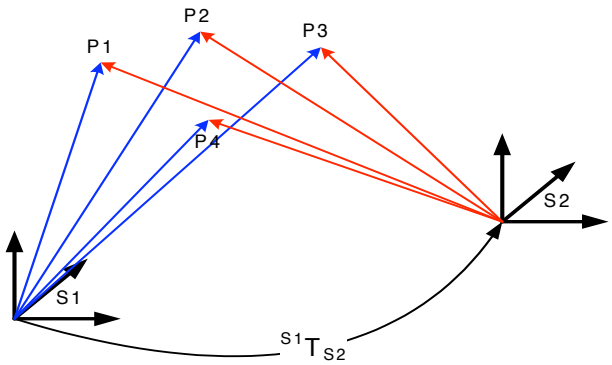

Figure 42: *Transformation graph robot and tracker points*

The transformation of one point given in robot coordinate system, to a point in tracker coordinate system can be described as follows:

$$^{rob}\vec{p}_n = ^{rob}\vec{T}_{trk} \cdot \vec{p}_{trk} \tag{53}$$

The most important part of the equation is now to calculate the transformation from robot to tracker. For this, the equation is modified, so that one side of the equation gets to zero:

$$^{rob}\vec{T}_{trk} \cdot \vec{p}_{trk} - ^{rob}\vec{p}_n = 0 \tag{54}$$

The transformation matrix $^{rob}\vec{T}_{trk}$ is consisting of a rotation matrix, described by $\vec{n}$, $\vec{o}$ and $\vec{a}$ and a translation, described by $\vec{t}$, converted to an orthonormal T-matrix [13]:

$$^{A}\vec{T}_{B} = \begin{pmatrix} n_x & o_x & a_x & t_x \\ n_y & o_y & a_y & t_y \\ n_z & o_z & a_z & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{55}$$

The rotation matrix part of the T-Matrix can be expressed in its trigonometric description, which creates the following equation:

$$\begin{pmatrix} \cos\gamma\sin\beta & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\gamma\sin\beta\cos\alpha + \sin\gamma\sin\alpha & t_x \\ \sin\gamma\cos\beta & \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha & \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha & t_y \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} ^{trk}x_n \\ ^{trk}y_n \\ ^{trk}z_n \\ 1 \end{pmatrix} - \begin{pmatrix} ^{rob}x_n \\ ^{rob}y_n \\ ^{rob}z_n \\ 1 \end{pmatrix} = \vec{0} \tag{56}$$

This equation is used to create an equation system, consisting of three equations including six variables for each point, so for each movement position of the robot. The third equation is not depending on γ:

$$\begin{aligned} &^{trk}x_n \cos\gamma\sin\beta + ^{trk}y_n \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha \\ &+ ^{trk}z_n \cos\gamma\sin\beta\cos\alpha + \sin\gamma\sin\alpha + t_x - ^{rob}x_n = 0 \end{aligned} \tag{57}$$

$$
\begin{aligned}
{}^{trk}x_n \sin\gamma\cos\beta + {}^{trk}y_n \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha + \\
{}^{trk}z_n \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha + t_y - {}^{rob}y_n = 0
\end{aligned} \tag{58}
$$

$$
\begin{aligned}
{}^{trk}x_n - \sin\beta + {}^{trk}y_n \cos\beta\sin\alpha \\
+ {}^{trk}z_n \cos\beta\cos\alpha + t_z - {}^{rob}z_n = 0
\end{aligned} \tag{59}
$$

For this reason, a minimum set of three points is necessary to calculate the transformation. In the experiments of this thesis a set of 10 measurement positions and with this overdetermined set of 30 equations was used for calculation to get better accuracy, see [8].

For solving the set of non-linear equations, the iterative Newton method [13] was used. The fundamental idea of this method is to find the minimum of a function, under usage of given starting conditions. For iteration, the tangent $t(x)$ on the function is used:

$$
t(x) = f(x_n) + f'(x_n)(x - x_n) \tag{60}
$$

The iteration rule is calculated by using the intersection of the tangent with the zero axis:

$$
f(x_n) + f'(x_n)(x - x_n) = 0 \tag{61}
$$

$$
\Leftrightarrow
$$

$$
x = x_n - \frac{f(x_n)}{f'(x_n)} \tag{62}
$$

Thus, the resulting iteration rule is:

$$
x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{63}
$$

The iterations are repeated until the breaking condition is reached. This breaking condition can be defined as a maximum value for x_{n+1} or as a maximum value for the function $f(x_n)$, which was used here.

$$|f(x_n)| <= \epsilon \tag{64}$$

Assigned to the 6-dimensional case under usage of 10 different measure positions follows:

$$\vec{f}(x_n) = \begin{bmatrix} f_{1,x}(\vec{x}_n) \\ f_{1,y}(\vec{x}_n) \\ f_{1,z}(\vec{x}_n) \\ \vdots \\ f_{10,y}(\vec{x}_n) \\ f_{10,z}(\vec{x}_n) \end{bmatrix} \qquad f'(\vec{x}_n) = J(\vec{x}_n) = \begin{bmatrix} \frac{\partial f_{1,x}(\vec{x_n})}{\partial t_x} & \frac{\partial f_{1,x}(\vec{x_n})}{\partial t_y} & \cdots & \frac{\partial f_{1,x}(\vec{x_n})}{\partial t_\gamma} \\ \frac{\partial f_{1,y}(\vec{x_n})}{\partial t_x} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial f_{10,x}(\vec{x_n})}{\partial t_x} & \cdots & \cdots & \frac{\partial f_{10,z}(\vec{x_n})}{\partial t_\gamma} \end{bmatrix} \tag{65}$$

Where J the iteration rule, which is used to solve the set of equations using the Newton iteration method, it can be described in matrix notification:

$$\vec{x}_{n+1} = \vec{x}_n - pinv(J(\vec{x}_n)) - \vec{f}(\vec{x}_n) \tag{66}$$

Where $pinv(J(\vec{x}_n))$ is the pseudo-inverse matrix of the Jacobian matrix $J(\vec{x}_n)$.

4.6.1 Interpolation of robot movements

One important step in the process of correction and improvement of the linear track is the mathematical identification of the track profile. In the last section it was explained, how the single identified coordinate systems of the sampling points on the track can be calculated.

Using this coordinate systems, the next approach is to get a mathematical function, describing the non-linear shape of the track. This can be used

to express the non-linearity of the track even between the sampling points.

The interpolating function [29], [54] should be adapted as good as possible to the real behavior of the track, because the resulting difference between mathematical description and real behaviour remains as an error component and is decreasing the resulting application accuracy.

Two different methods for description of the track were used and analysed in this thesis and tested with an own developed MATLAB program [29], [60], [47]. The first method is to find one function for interpolation of all sampling positions. The second method is to find an interpolation function composed of multiple functions describing the track behavior between each two sampling points.

The calculation of the interpolation function was implemented in a separate program offline to the application. The used data are real measurement data from a robot movement, measured with a Leica laser tracker.

Additionally to the robots linear track movements, different geometries of robot movements were analysed. For the interpolation of 3D-movements, each direction was separately interpolated. The basic idea of all three methods was to find a continous function $g(x)$ which approximates a given tabulated function $f(x)$ in the way that:

$$(f(x) - g(x))^2 < \epsilon, \quad \epsilon > 0 \tag{67}$$

4.6.2 Applied interpolation methods

Polynomial interpolation

The basic idea of the polynomial interpolation method is to find the interpolating function $g(x)$ by construction of one polynomial function with the degree of N. This polynomial function is fix for one set of sampling points. The polynomial of degree N, which interpolates a function $f(x)$ given through N tabulated values [14] is represented by:

$$p_n(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \tag{68}$$

For the calculation of $p_n(x)$ there are different methods possible to use, such as Newton, Bessel, Lagrange or Stirling [13]. All this methods are well known and can be chosen considering the needs and contraints of the current application. The interpolation results, used in this thesis were created by the least-square method [12].

Trigonometric interpolation

The trigonometric interpolation method is based on an interpolation function consisting of trigonometric components [14]. These components are calculated individually and composed to the function $t_n(x)$:

$$t_n(x) = a_0 + \sum_{j-1}^{n} a_j \cos(jx) + \sum_{j-1}^{n} b_j \sin(jx) \tag{69}$$

For an efficient calculation of the trigonometric components of the interpolating function $t_n(x)$ the FFT-algorithm was used [14].

Cubic Spline interpolation

The basic difference of the cubic spline interpolation method compared to the polynomial and trigonometric interpolation methods is the split-up-behavior of this function. The interpolating function using this method is not representing the whole interpolation area, but only the area between two sampling points.

The cubic spline interpolation is based on the theory to interpolate a tabulated function in sub-parts by calculation of cubic spline elements between the sampling points of the given tabulated function, thus:

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \tag{70}$$

One great advantage of this method is the flexibility in the number of elements. The effort for calculation of the interpolating function is linear increasing with the number of sampling points of the tabulated function.

Another positive aspect for the choosing of cubic spline interpolation is the

two-times continuously differentiable ability of the splines. This behavior is very close to the behaviour of deforming metal parts and with this excellent for the interpolation of linear tracks.

4.6.3 Interpolation tests

As a test case for interpolation methods for industrial robot movements, different geometries were tested in this thesis. Each geometry covers certain movement and accuracy aspects of the robot. They are part of a test path integrated in the european standard ISO 9283 [2] in the so-called "optional test path".

Tested geometries

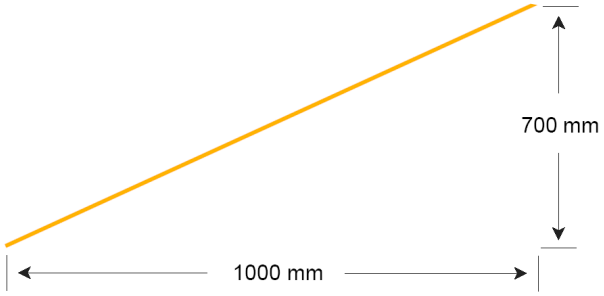

Figure 43: Straight line movement

As a simple but mostly used movement, a straight line geometry (figure 43) was tested. For this, the used robot was programmed with a linear movement from point A to point B with movement distances of 1000 mm and 700 mm in each direction.

As a more difficult task, in step two there is a circular movement part integrated into the straight line movement (see figure 44). This represents applications like sealing of tire frames in automotive production.

This geometry is more difficult for the robot to move, because it represents edges and circular parts. Even for the interpolation it is more difficult to find an interpolation function with low difference values from the real measurements.

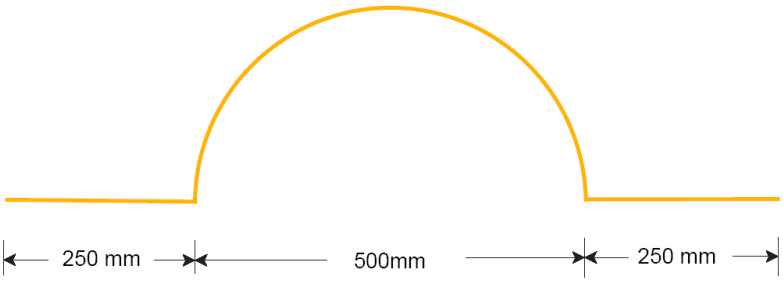

Figure 44: *Lin-Circ-Lin movement*

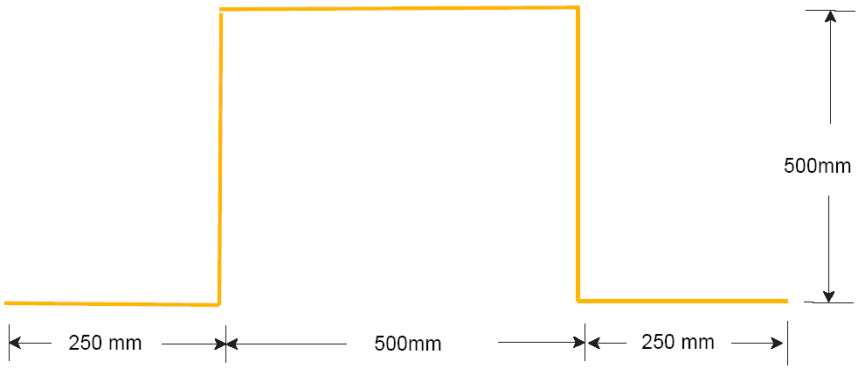

Figure 45: *Rectangular movement*

The third and most complex movement geometry is done by an rectangular movement (figure 45). This movement is on the one hand difficult to move for the robot, on the other hand even difficult to interpolate exactly, caused by the several edges in the movement path.

Interpolation results

For calculation of the interpolation quality, the tabulated function was interpolated, using the three mentioned interpolation methods. After that, a comparison of the interpolation function with a second measurement run was done. This measurement run was taken at a very low movement speed of the robot and a high sampling rate of the measurement system.

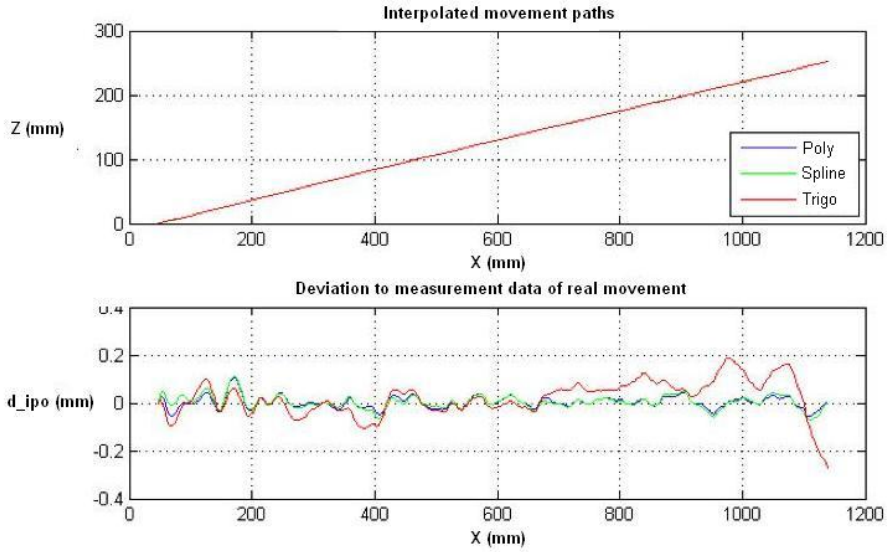

Figure 46: *Interpolation of straight line movement*

The figures 46 and 47 are showing the interpolated path and the error d_{ipo} between real movement and interpolation function for the three interpolating functions polynomial (blue), cubic spline (green) and trigonometric interpolation. The interpolated movement path is shown in the top of figures 46 and 47, the resulting deviation to the high-accurate measurement data of the slow run is shown in each bottom figure.

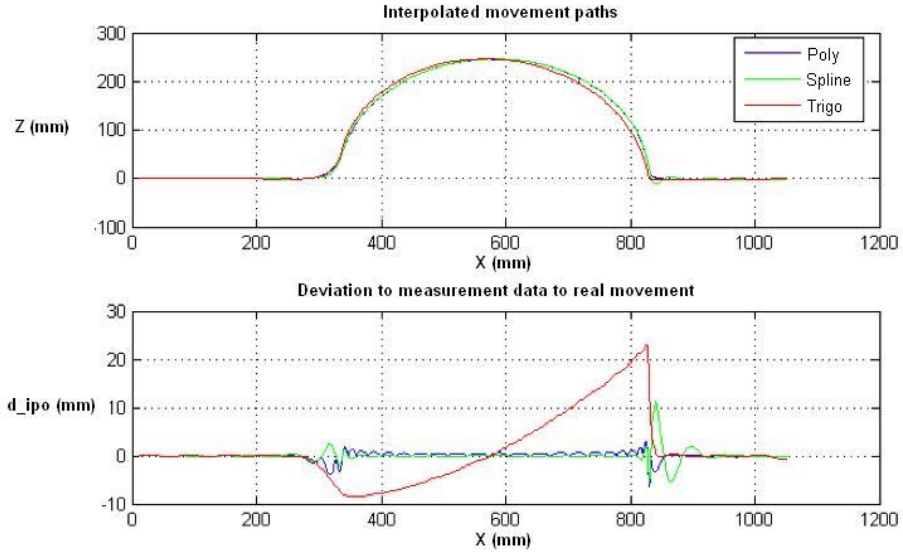

Figure 47: *Interpolation of lin-circ-lin movement*

In figure 48 there is only the trigonometric interpolation function shown. Applied on rectangular movements, the interpolation results of the trigonometric interpolation is much better.

Considering the movements geometries, which are directly applicable to the real track deformations, the focus is on linear and curve-shaped forms. With this, the cubic spline interpolation was the most accurate and easy-to-calculate interpolation method. For this reason, the cubic spline interpolation was chosen to be implemented in the correction algorithm and with this even into the correction program.

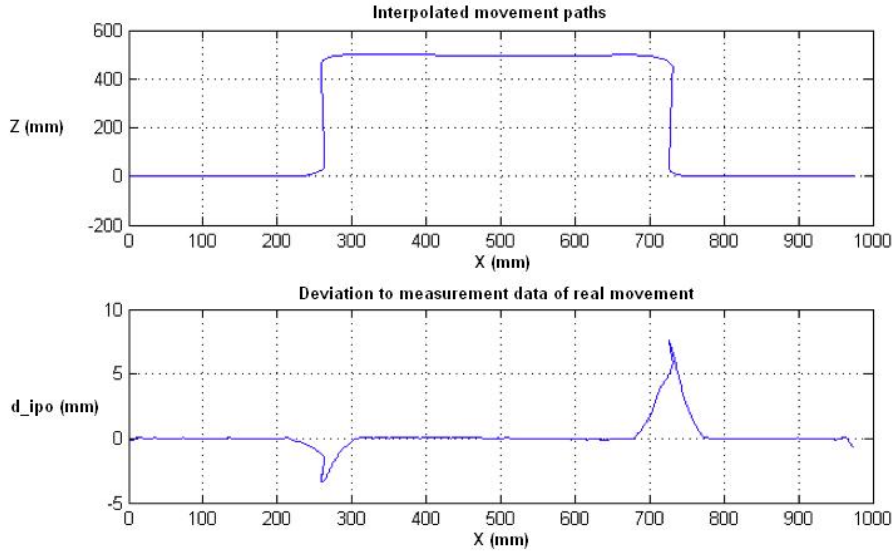

Figure 48: *Trigonometric Interpolation of rectangle line movement*

4.6.4 Implementation of cubic spline interpolation

The origin for the interpolation is a given tabulated function $F(x_n)$ defined by the calculated coordinated systems $\vec{T}_i$ of the track.

This 6-dimensional description of the n sampling points is divided into 6 different 1-dimensional functions in $x, y, z, \alpha, \beta, \gamma$. For example, the function for z is given by:

$$R_z(x_n) = R_0(x_0, y_0), R_1(x_1, y_1), ..., R_n(x_{n-1}, y_{n-1}) \tag{71}$$

One of the basic advantages of the interpolation of R_z using cubic splines is the low effort for increasing the number of sampling points, due to the calculation of sub-functions - is this case called splines - between two sampling positions.

This splines, notated by S_i are combining the sampling points to each other and satisfying pre-defined constraints. Due to this constraints the characteristic of the interpolation function is smooth and close to the real geometrical behavior of the linear track.

The given constraints are defined as follows:

- The interpolating function $S(x)$ hits every sampling point exactly.
- Two splines of $S(x)$ are hitting each other exactly in the sampling points.
- The splines of $S(x)$ are due to their cubic order two times steady differentiable.

At all, for calculation of the interpolating function $S(x)$ for a given tabulated function $R_z(x_n)$ a calculation of $n-1$ splines is necessary. This can be done in an efficient algorithmic way, which is explained in the following. All this calculations were done for each degree of freedom separately.

Non-parametric cubic spline function

As a prediction for the interpolation using non-parametric cubic splines, the given tabulated points of the function *R(x)* have to be represented in an ordered way for each degree of freedom [14], thus:

$$a := x_0 < x_1 < \ldots < x_n =: b \tag{72}$$

This is always given, if the track profile was measured sequentially along the movement direction of the track.
As a second prediction, the resulting interpolating function $S(x)$ has to be 2-times differentiable in every point. As mentioned before, the cubic splines of the the interpolation function combined together result in $S(x)$:

$$S_1(x), S_2(x), \ldots, S_n(x) := S(x) \tag{73}$$

The splines $S_i(x)$ are represented by a ploynomial with degree of three and a set of four unknown variables:

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \tag{74}$$

$$\text{with} \qquad x \in [x_i, x_{i+1}], \quad a_i, b_i, c_i, d_i \in \mathbf{R}, \quad i = 0(1)n$$

The function $S(x)$ can now be constructed from $n-1$ spline functions $S_i(x)$ where n represents the number of tabulated values in $R(x)$. For every $S_i(x)$ there are 4 different unknown coefficients to calculate, which gives in total an amount of $4(n-1)$ coefficients.

The needed equation set for calculation is given via function constraints of the splines. With the prediction to be 2-times differentiable, the constraints to hit in every point and to fit to each other, for the splines follows:

$$S_i(x_i) = y_i, \qquad i = 0(1)n \tag{75}$$
$$S_i(x_i) = S_{i-1}(x_i), \quad i = 1(1)n \tag{76}$$
$$S_i'(x_i) = S_{i-1}'(x_i), \quad i = 1(1)n \tag{77}$$
$$S_i''(x_i) = S_{i-1}''(x_i), \quad i = 1(1)n \tag{78}$$

The corresponding derivative to *S(x)* can also be calculated using the polynomial representations, thus:

$$S_i'(x_i) = b_i + 2c_i(x - x_i) + 3d_i(x - x_i)^2 \tag{79}$$

and

$$S_i''(x_i) = 2c_i + 6d_i(x - x_i) \tag{80}$$

Setting $h_i = x_{i+1} - x_i$ as sampling difference depending on the used sampling rate during scan measurements of the linear track, gives an algorithm to calculate the unknown coefficients of the splines $S_i(x)$:

$$a_i = y_i \tag{81}$$

$$b_i = \frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}(c_{i+1} + 2c_i), \quad i = 0(1)n - 1 \tag{82}$$

$$d_i = \frac{1}{3h_i}(c_{i+1} - c_i), \quad i = 0(1)n - 1 \tag{83}$$

The needed coefficients c_i for equations 82 and 83 are taken from a linear set of equations represented by a matrix multiplication:

$$\vec{A} \cdot \vec{c} = \vec{a} \tag{84}$$

This matrix is constructed by a set of equations which are used to identify c_i depending on the current index i:

$i = 1$:

$$2(h_0 + h_1)c_1 + h_1c_2 = 3\frac{y_2 - y_1}{h_1} - 3\frac{y_1 - y_0}{h_0} - h_0c_0 \tag{85}$$

$i = 2(1)n, n >= 4$:

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_ic_{i+1} = 3\frac{y_{i+1} - y_i}{h_i} - 3\frac{y_i - y_{i-1}}{h_{i-1}} \tag{86}$$

$i = n-1, n >= 3$:

$$h_{n-2}c_{n-2} + 2(h_{n-2} + h_{n-1})c_{n-1} = 3\frac{y_n - y_{n-1}}{hn-1} - 3\frac{y_{n-1} - y_{n-2}}{hn-2} - h_{n-1}c_n \quad (87)$$

The matrix **A**, **c** and **a** which describe the complete equation set for the calculation of the c_i-parameter can be expressed by:

$$\mathbf{A} = \begin{pmatrix} 2(h_0+h_1) & h_1 & 0 & \dots & 0 \\ h_1 & 2(h_1+h_2) & h_2 & \dots & 0 \\ 0 & h_2 & 2(h_2+h_3) & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{pmatrix} \quad (88)$$

$$\mathbf{c} = \begin{pmatrix} c_1 \\ c_2 \\ \dots \\ c_{n-2} \\ c_{n-1} \end{pmatrix} \quad (89)$$

$$\mathbf{a} = \begin{pmatrix} 3 \cdot \frac{y_2-y_1}{h_1} - 3 \cdot \frac{y_1-y_0}{h_0} - h_0 c_0 \\ 3 \cdot \frac{y_3-y_2}{h_2} - 3 \cdot \frac{y_2-y_1}{h_1} \\ 3 \cdot \frac{y_4-y_3}{h_3} - 3 \cdot \frac{y_3-y_2}{h_2} \\ \dots \\ 3 \cdot \frac{y_n-y_{n-1}}{h_{n-1}} - 3 \cdot \frac{y_{n-1}-y_{n-2}}{h_{n-2}} - h_{n-1}c_n \end{pmatrix} \quad (90)$$

The special structure of the matrix **A** brings an advantage to the solving algorithm of the equation set. It is tridiagonal, symmetric and strongly diagonal dominat with only positive elements. With this predictions, **A** is proofed to be invertable.

With the ability of **A** being invertible, the equation set can be solved using an appropriate method, for example the Gauss or Cholesky algorithm [12]. The missing parameters c_0 and c_n can be calculated using the boundary conditions.

4.7 Correction of the linear track

To decrease the positional deflections components of the robots TCP which are caused by the linear tracks geometrical deflections, the track accuracy can be improved in two different ways.

4.7.1 Manual adjustment using track analysis data

Using the continuos description of the track profile after finding the interpolating functions $S_p(x)$, a manual adjustment of the track can be done. For this, the adjustment screws of the track can be used, to improve the geometrical shape of the track rails in z- and y - direction (assuming x as movement-direction)

The interpolation function delivers a continuos description for each degree of freedom in an arbitrary position on the track:

$$S_p(x) \rightarrow S_x(x), S_y(x), S_z(x), S_\alpha(x), S_\beta(x), S_\gamma(x) \tag{91}$$

To achieve an optimal position and shape of the track rails, the resulting accuracy is depending on the linearity of the track rails. The resulting direction of the rails after assuring good linearity behavior, can be compensated through the alignment of the robot.

Thus, the difference between the interpolation function and a regression line through the sampling frames of the track defines the resulting compensation offset for manual adapting. For one example degree of freedom, in z:

$$\Delta z_n = S_n(x) - b_{0,z} - x \cdot m_z \tag{92}$$

Where $b_{0,z}$ is the y-axis value and m_z the slope value of the regression line for z.

A sample measurement of a linear track for a middle-size sealing robot shows gives an overview to the occuring differences (table 4).

	x	**y**	**z**	α	β	γ
Pos1	0	0	0	0	0	0
Pos2	1999.566	0.682	-0.428	0.0100	0.0002	0.0056
Pos3	3999.563	1.718	-0.651	0.0119	0.0030	0.0116
Pos4	5999.005	3.412	-0.852	359.9981	359.9954	0.0336
Pos5	7998.801	5.679	-0.657	0.0205	359.9968	0.0593

Table 4: *Detected coordinate frames in sampling positions*

The linear track was measured in 5 different skid positions with the same 10 different TCP positions of the robot. Corresponding to that, there are 5 different frames identified, which are depicted as Pos1 to Pos 5 in table 4.

The resulting regression line in relation to the identified z-positons of the the coordinate frames is shown in figure 49. Based on the fact, that the measurement system is exactly leveled, it can be seen that the linear track is going down in z while moving the robot in positive x-direction along the track.

The position of the coordinate frames in z with relation to the first coordinate systems can also be seen in figure 49. It shows, that the track is slightly arcuated if considering only z-direction.

The more important part of the position information is given by figure 50. The absolute error between one arbitrary coordinate system of a sampling point, compared to the first one is an useful practical value to see the geometry of the track in total. But, as mentioned before, the main track direction is identified by the alignment and with the proper alignment vector integrated into the robot control, even compensated.

The positional deviations to this regression line as they are shown in figure 50 have to be compensated in a different way, in this case manually by

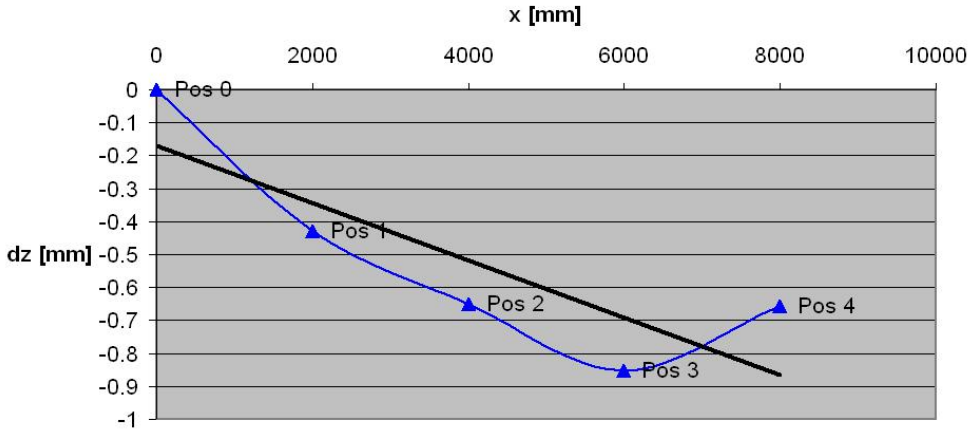

Figure 49: *Sampling points and regression line for z-coordinate*

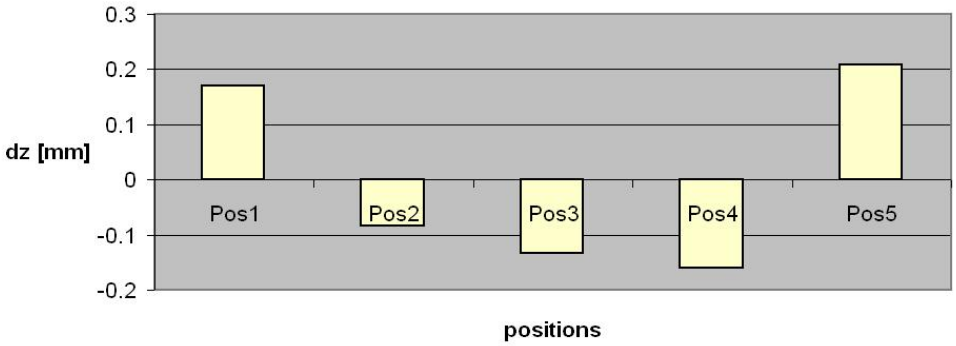

Figure 50: *Differences to best-fit line in z-direction*

using the adjustment screws of the linear track.

One crucial aspect of adjusting the linear track manually is, that due to the coupled system, the adjustment of one part of the linear track is influencing another part of the track.

The translational parts of the deviation vector is intuitively adjustable, but the coupled rotational deviations are hard to control. For this reason, a special method for depicting the rotational influence and direction was implemented in this thesis.

The so-called twist diagram is a permutation of the rotational influence into a translational influence (figure 51).

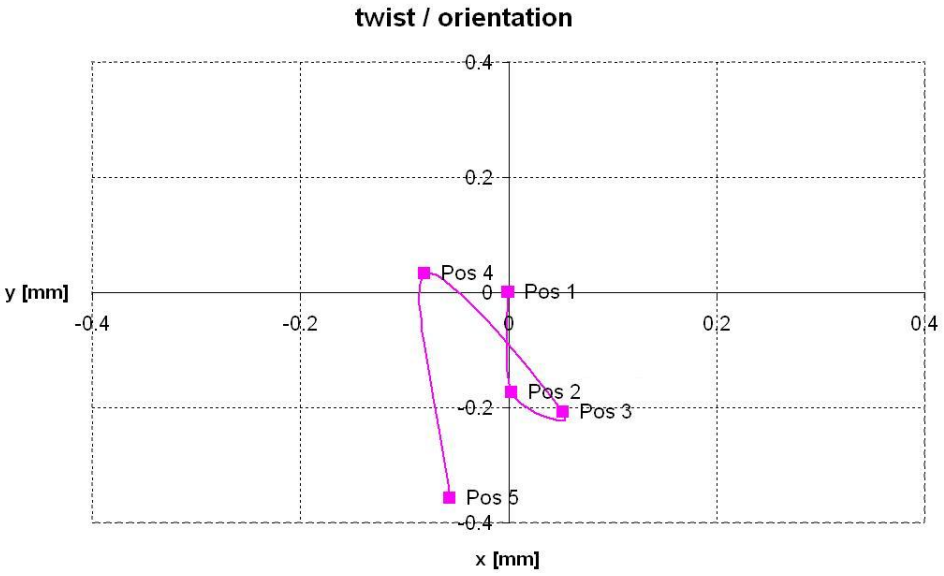

Figure 51: *Twist diagram*

The created curve is the 2D-projection of 5 different sampling positions (Pos 1 - Pos 5) on the linear track, each measured in 3D-coordinates. The projection is done into the x,y-plane. Amplified by a lever of 1 meter in $+z$-direction in robot base, the deviations to the start position Pos 1 are detectable in the 2D-plane diagram (figure 51). A rotational difference of 0.1° for example, is causing in a translational deviation of 1.75 mm in the diagram.

Using this diagram, it is possible to use the adjustment screws of the linear track to apply a proper position and shape to the linear track rails and with this to improve the resulting accuracy of the track without correction by software.

4.7.2 Theorectical compensation of the linear track

Without adaptation of the real geometry and position of the linear track it is possible to improve the application accuracy by correction of the robot movements theoretically. With this, the robots current skid position on the track is used to calculate a 6-dimensional correction vector to compensate the linear track error influences.

This procedure can be on the one hand used to improve the accuracy of one offline constructed robot program, but on the other hand also for adaptation to other robots.

The adaptation of one robot program to another robot on a different linear track is based on the different geometries of their linear tracks a sensitive process. Subchapter 8.1 is dealing with the demands of this tasks.

Modification of robot program

The basic idea of theoretical compensation of linear tracks is an integrated correction offset for the robot program. Depending on the skid position, this correction value differs. In figure 52 are two different robot positions *RobPos1* and *RobPos n* depicted.

In *RobPos 1*, the robot uses a workobject transformation $WO1$. The desired target position of the robot in this linear track position is depicted as $T_{offline}$.

If the robot now moves along the linear track to another position *RobPos n*, its translational position and - more crucial - its orientation will due to the linear track geometry not be the same as in position *RobPos1*. To hit the same point at the workobject the programed position now should be T_{def} which is different to $T_{offline}$ which was used before (the translational shift caused by the linear track movement from *RobPos1* to *RobPosn* is not focused because it is integrated in the program commands).

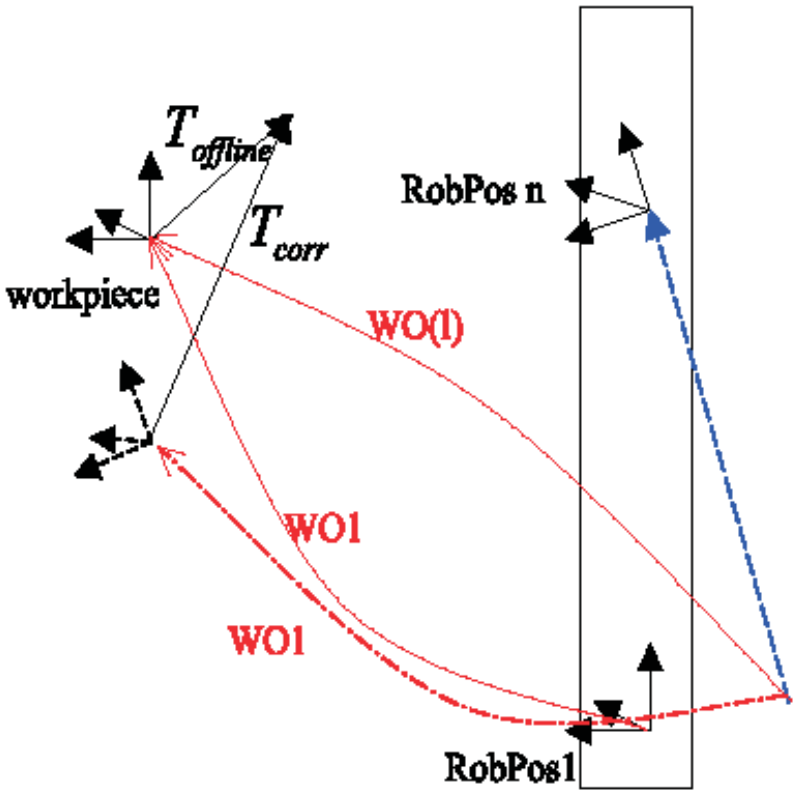

Figure 52: *Correction offset*

To use the offline robot program with the programed point $T_{offline}$ even at *RobPos n*, an adapted transformation $WO(l)$ depending on the current skid position is calculated, which contains the transformation $WO1$ and also a correction transformation depending on the skid position on the track. This correction transformation is applied to the robot program point $T_{offline}$, thus:

$$T_{corr} = (WO1)^{-1} \cdot WO(l) \cdot T_{offline} \tag{93}$$

After modification of all offline generated robot program points, the current program is adapted to the linear track real profile. With this, the deflections caused by the non-linear geometry of the track are compensated individually

in each position on the track.

Online correction during robot movement

The basic step of correcting the robot movements is done by offline modification of the robot program. This can be done by special automated software in which the prior measured and calculated linear track profiles are included.

A more flexible way of correction is the modification of the robots TCP position during robot movements. For this, an appropriate interface control is needed, to assure a high-speed correction of the current TCP position.

With this interface, a correction value for the current TCP position of the robot can be processed in the interpolation cycle of the robot control. The corresponding correction value can be calculated by an autonomous software from the current sensor data of the linear track position encoder.

This software, needed for the calculation and interchanging of the correction parameters to the robot control was during this work implemented as a prototype working with ABB robots.

4.8 Software module for automated movement control

The prototype software for the automated correction of the robots TCP is a program developed in C++. It is consisting of a graphical user interface based on MFC (Microsoft Foundation Class) for easy handling.

Figure 53 is presenting the special module structure of the software system.

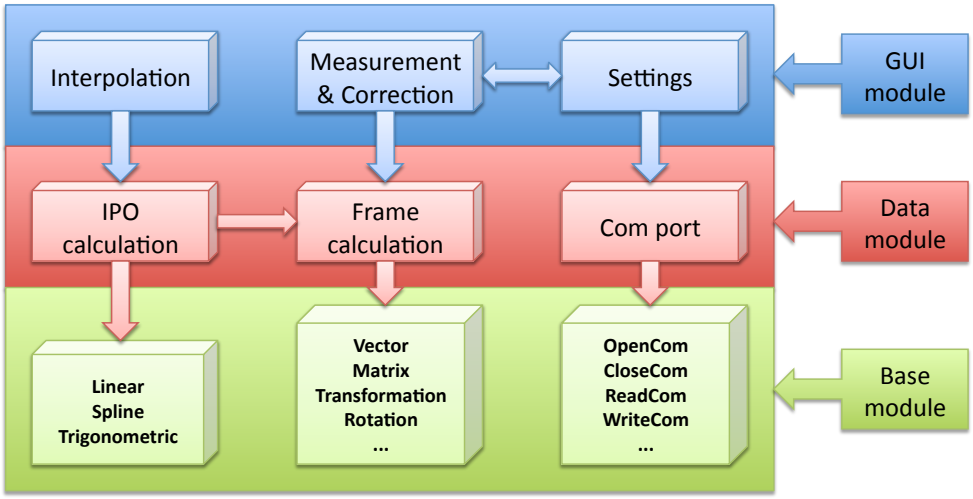

Figure 53: *Module structure of online correction program*

Divided into three different sections, the base module of the software is responsible for basic calculation and communication routines. This routines provide functionalities like the calculation of the implemented spline functions, the appropriate matrices and vectors or communication tasks like R/W-operations on the robot interface.

The data module is on a higher logical level and is collecting and using the calculated data from the basis module. In this part of the software, the complete interpolation results and coordinate systems are calculated or the robot interface is globally controlled.

On the top, the GUI module is providing the user interface functionality including the configuration of the software settings like interpolation or connection parameters.

4.8.1 Software functionalities

The graphical user interface of the software is consisting of a dialog based window like shown in figure 54.

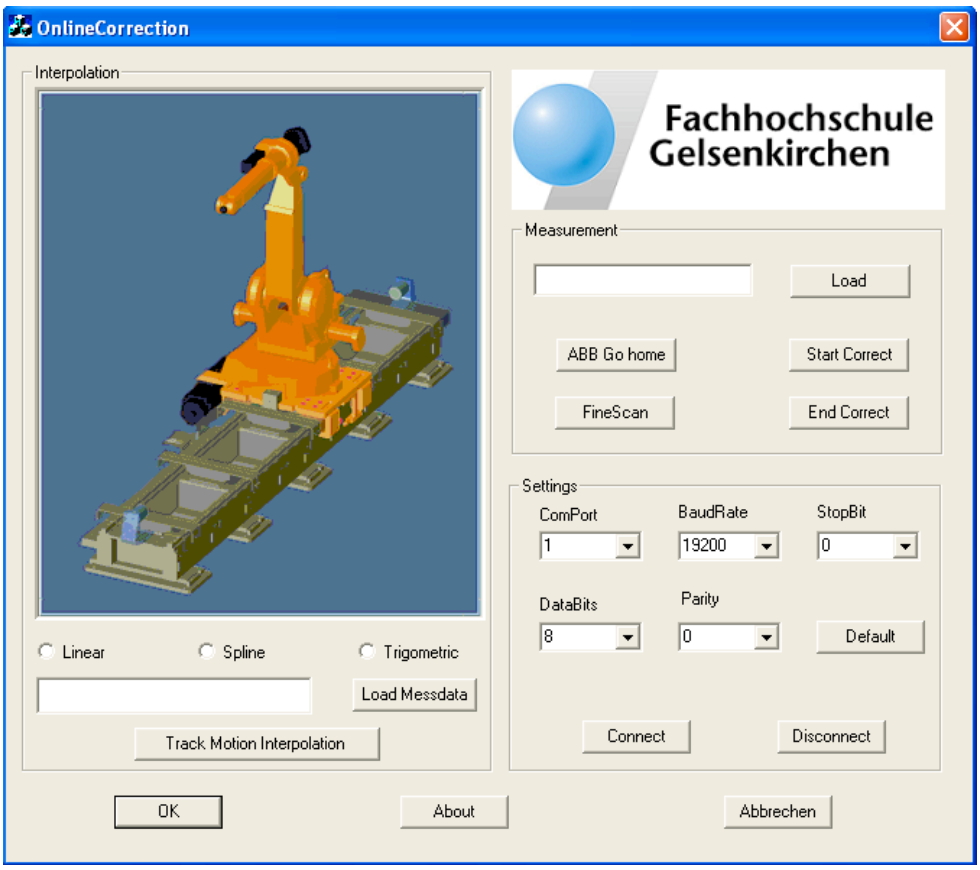

Figure 54: *Graphical user interface of software prototype*

The program is structured into three parts: interpolation, measurement and settings. Every single block is responsible for a different part of the program where the user can take action in changing parameters or calculation predictions.

Interpolation

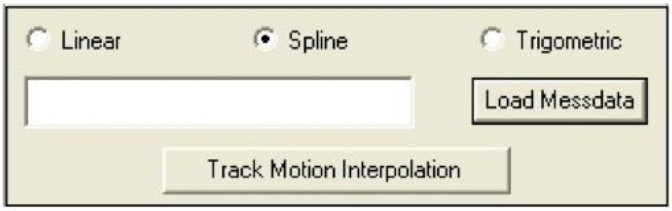

Figure 55: *Interpolation function block*

Three different possible interpolation methods are available for the user (figure 55). For easy changing of the offered interpolation methods, each of them is implemented in an own C++ class. Using the button "Load Messdata", the prior measured scan data of the linear track can be opened and used for the interpolation. After opening the measurement data, the "Track Motion Interpolation" button can be used to calculate the interpolation function for the track. Figure 56 shows an example for a cubic spline interpolation on real linear track scan data generated by the software.

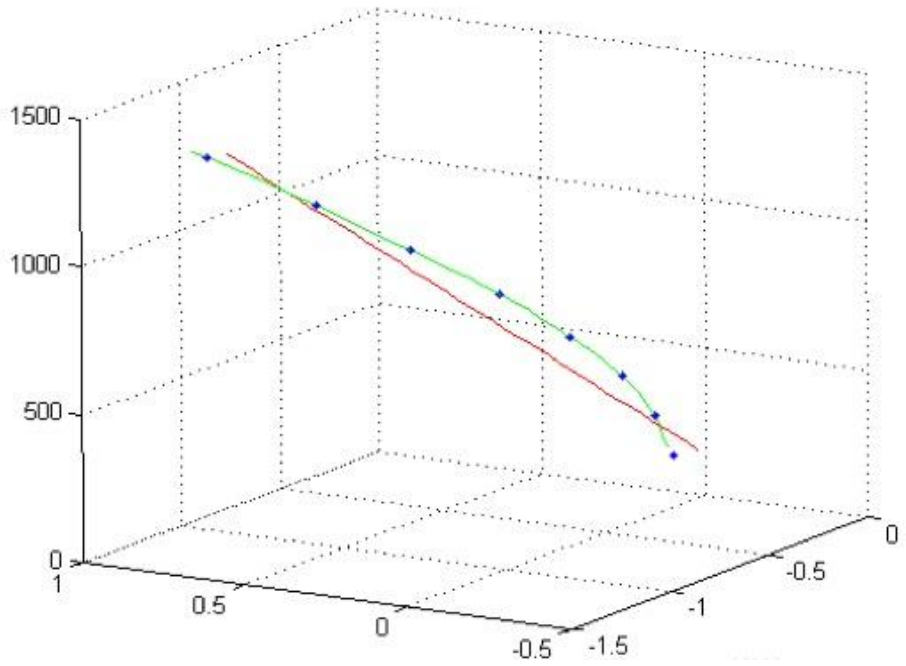

Figure 56: *Interpolation result*

Measurement

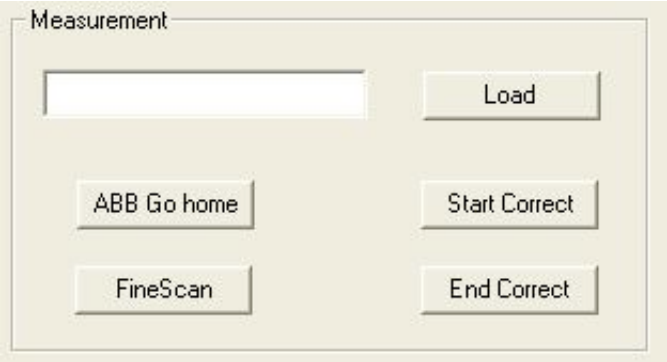

Figure 57: *Measurement function block*

This block is reflecting the program structure for the automated position control of the robots TCP (figure 57). One of this is the ABB "Go home" function which is used to place the robot to an exact start position before every run.

Using the method "FineScan" offers the user an ability to move the robot at a very low movement velocity and without correction of the TCP along the track. The resulting measurement data can afterwards be used for calculation or for a comparison of the interpolation to the real measured test data from this low speed run.

After connection with the robot using the appropriate button and loading the low speed measurement data, the automatic correction mode can be started.

Settings

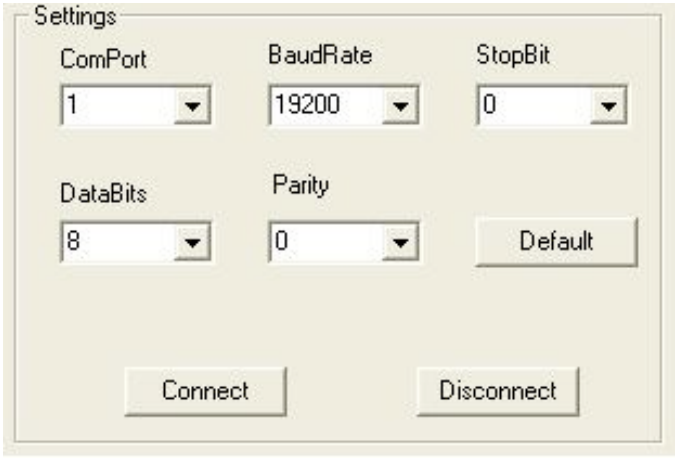

Figure 58: *Settings function block*

For the communication between robot control and the user PC for the prototype software, a serial connection was used (figure 58). To configure this serial connection, it is possible for the user to change the connection parameters using the *Settings* function block of the software. With this, it is possible to adapt the robot and PC settings to each other for an optimal communication rate between both partners robot and controlling PC.

4.8.2 Robot program software structure

For the automated correction, the software module which is working on the user PC is not sufficient. For controlling the robot, it is necessary to run a special robot program, which is corresponding with the PC program on the user side.

This, on the robot control running program, is receiving and sending commands using the serial interface between robot and user PC. The flowchart in figure 59 illustrates the process of the robot program.
Before the program starts, the serial connection has to be established. This is done by using the according button on the teach panel using "TPReadFK". Following on this, the "command" routine is sending commands generated by function buttons which can be selected in the user software. On sending an "end connection command (ENDCO)" the program is disconnecting from or to serial port.

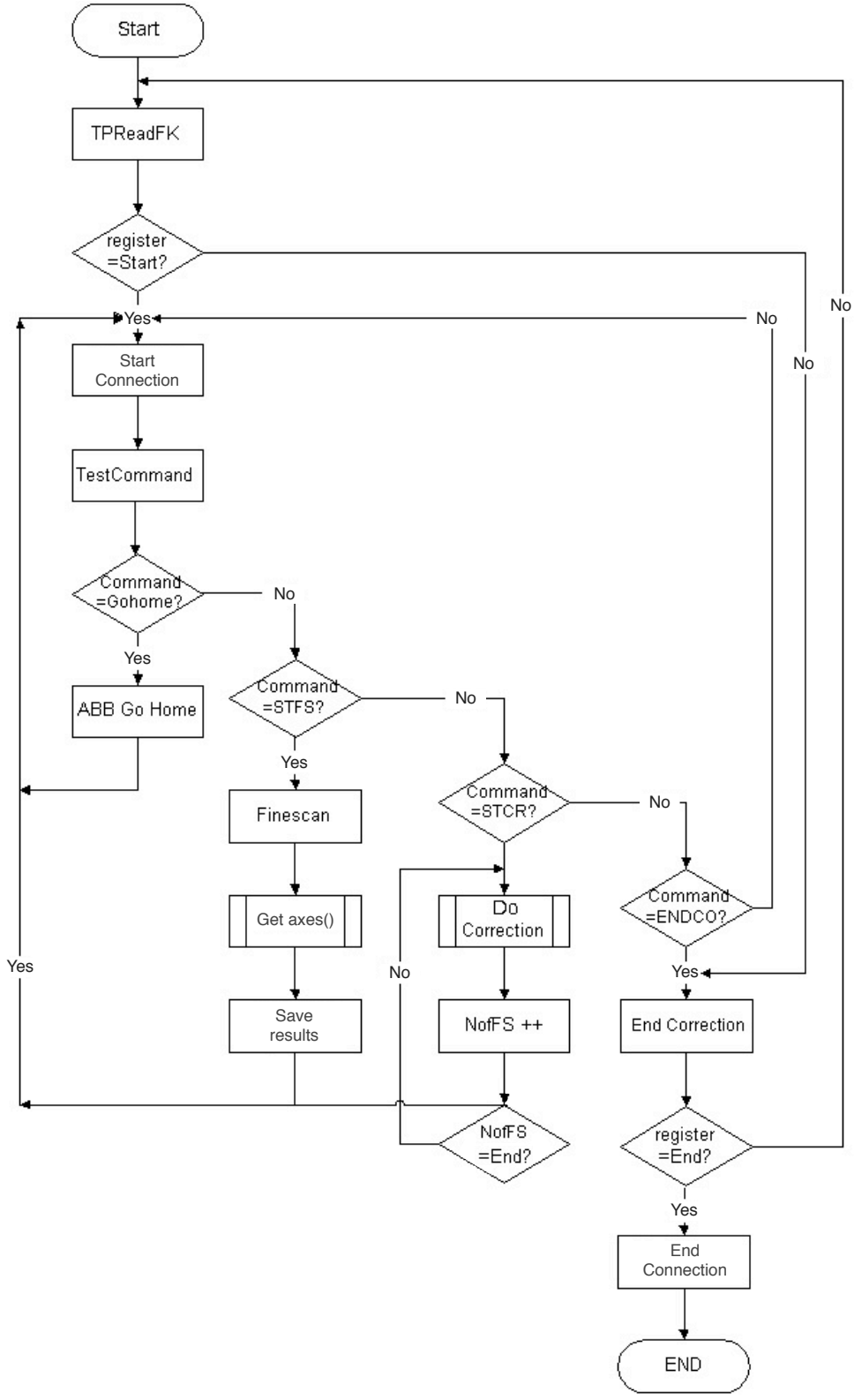

Figure 59: *Robot program software structure*

4.9 Error analysis

The interpolation, using the cubic splines can never be perfectly adapted to the real behavior of the linear track. For this, a residual error is existing, which is estimated in the following section.

One aspect of the mathematical description of the robot movement along the linear track, is the interpolation of discrete measured points, which describe the robot movement. One interesting part is the maximum possible deviation between the linear and the cubic spline interpolated function.

For this, measurement results from a high-accuracy measurement of the track are taken to deliver predictions about possible diffusion and deviations of the measurement values. The goal is, to take a quantitative prediction about the resulting error of the interpolation.

Geometrical derivation

With a given cubic spline function $S(x)$, defined on $x \in [x_a, x_b]$, the perpendicular distance h between a straight line $G(x)$ from x_a to x_b and $S(x)$ is needed (figure 60).

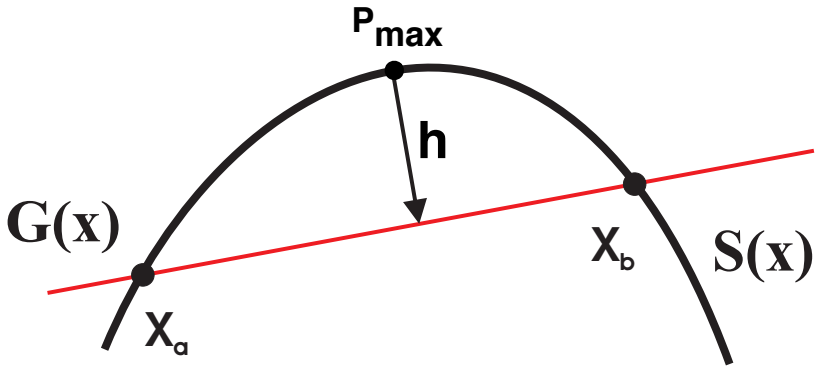

Figure 60: *Interpolation of two points using cubic splines*

Based on the attributes of cubic spline functions, the resulting interpolation curve can be defined by:

$$S(x) = a + b(x - x_a) + c(x - x_a)^2 + d(x - x_a)^3 \tag{94}$$

In dependency to the point coordinates of x_a and x_b, the resulting spline can take different shapes (figure 61):

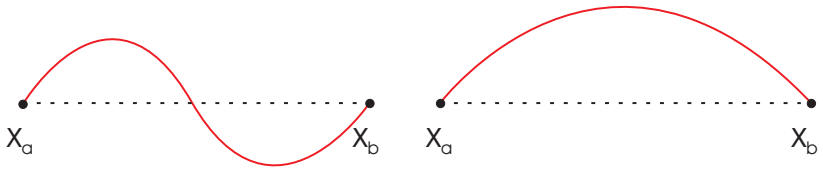

Figure 61: *Possible geometrical shapes of* $S(x)$

The next question is, how the geometrical shape of the interpolation function has to be like, for getting a maximum distance between cubic spline $S(x)$ and straight line $G(x)$.

As a prediction should be, that in the point with the most bending of the curve, the maximum distance is located (figure 61).

Calculation of the maximum clearance

Like shown in figure 60, a cubic spline $S(x)$ and a straight line $G(x)$ between two points A and B are given. We must find the maximum perpendicular distance d between $S(x)$ and $G(x)$.

The corresponding function descriptions are equation 94 and:

$$G(x) = \frac{y_b - y_a}{x_b - x_a} \cdot x + \frac{x_b y_a - x_a y_b}{x_b - x_a} \tag{95}$$

With a parallel shift of $G(x)$ by the distance d, the straight line $G(x)$ is touching the spline $S(x)$ in exactly one point P_{max}. This point P_{max} on the cubic spline has got the maximum distance d to $G(x)$.

To find P_{max} in a calculative way, a point on the cubic spline has to be detected, which has got the same gradient like in P_{max}. In this example, the

result is exact, because $S(x)$ has got no turn tangent between x_a and x_b.

It follows:

$$S'(x_{max}) = m = \frac{y_b - y_a}{x_b - x_a} \tag{96}$$

Thus:

$$S'(x_{max}) = b + 2c(x_{max} - x_a) + 3d(x_{max} - x_a)^2 = \frac{y_b - y_a}{x_b - x_a} = m \tag{97}$$

$$\Leftrightarrow \quad b + 2cx_{max} - 2cx_a + 3dx_{max}^2 + 3dx_a^2 = m + 6dx_{max}x_a \tag{98}$$

$$\begin{aligned} \Leftrightarrow \quad & x_{max}^2 + 2\frac{c - 3d}{3d} \cdot x_a \cdot x_{max} + \frac{c - 3d}{3d} \cdot x_a{}^2 \\ & = m + \frac{c - 3d}{3d} \cdot x_a{}^2 + 2cx_a - 3dx_a^2 - b \end{aligned} \tag{99}$$

$$\Leftrightarrow \quad \left(x_{max} + \frac{c - 3d}{3d} \cdot x_a\right)^2 = \frac{m - b + 2cx_a}{3d} - x_a^2 + \frac{c - 3d}{3d} \cdot x_a{}^2 \tag{100}$$

$$\Leftrightarrow \quad x_{1,2} = -\frac{c - 3d}{3d} \cdot x_a \pm \sqrt{\frac{m - b + 2cx_a}{3d} - x_a^2 + \frac{c - 3d}{3d} \cdot x_a{}^2}, \quad d \neq 0 \tag{101}$$

The solution for x_{max} shows explicitly, that if the parameter d of the spline function is non-zero, two maximum deviations can be found.

4.10 Relevance of the factor d

In the following it will be shown, that the maximum deviations will be more, if the factor d gets to zero.

For the construction of a cubic spline from point X_a to point X_b it is an important consideration which border derivation are used in the calculation. With varying differentiations in the points X_a and X_b, the curve will take different bendings.

The maximum bending of $S(x)$ is reached with a maximum gradient difference between X_a and X_b. This has got a maximum if the gradient m_a in X_a and the gradient m_b in the point X_b is like:

$$m_a = -m_b \tag{102}$$

The maximum possible difference between two sampling points of the track measurement in one coordinate is influencing the maximum gradients. A bigger distance between X_a and X_b is increasing the resulting difference between $S(x)$ and $G(x)$.

For a maximum clearance from $S(x)$ to $G(x)$, no turning point will be existing between X_a and X_b. The avoidance of a turning point results always in a positive factor d of the spline function [15].

4.11 Example of the calculation with given border derivations

Given is a set of measured sampling points (x_i, y_i), $i = 0(1)3$ with

i	0	1	2	3
x_i	0	100	200	300
y_i	-0.25	0.25	0.25	-0.25

The interpolating cubic spline function $S(x)$ has to be calculated using given border derivations:

$$S'(x_0) = \frac{y_1 - y_0}{x_1 - x_0} = m \quad S'(x_3) = \frac{y_3 - y_2}{x_3 - x_2} = m' \tag{103}$$

Coefficient calculation
The first step is the calculation of the coefficients c_i, the only non-trivial coefficients, which are not depending on other coefficients. The a_i coefficients are represented by the y_i measurement values of the sampling points.

The resulting coefficients c_0 to c_3 are represented by:

$$c_0 = \frac{1}{2h}\left(\frac{1,5}{h} - 3m - hc_1\right) \tag{104}$$

$$c_1 = -\frac{1}{3}\frac{1}{h^2} = -\frac{1}{3}\cdot 10^{-4} \tag{105}$$

$$c_2 = -\frac{1}{3}\frac{1}{h^2} = -\frac{1}{3}\cdot 10^{-4} \tag{106}$$

$$c_3 = -\frac{1}{2h}\left(-\frac{1,5}{h} - 3m' + hc_2\right) \tag{107}$$

The other parameters of the cubic spline functions $S_0(x)$, $S_1(x)$ and $S_2(x)$ can be calculated by simple insertion in the corresponding formulas [14].

Resulting are the following functions:

$$\begin{aligned} S_0(x) &= -0.25 + 5\cdot 10^{-3}(x-x_0) + 0.0166\cdot 10^{-3}(x-x_0)^2 \\ &\quad -0.166\cdot 10^{-6}(x-x_0)^3 \end{aligned} \tag{108}$$

$$S_1(x) = 0.25 + 3.33\cdot 10^{-3}(x-x_0) - 0.033\cdot 10^{-3}(x-x_0)^2 \tag{109}$$

$$\begin{aligned} S_2(x) &= 0.25 - 3.33\cdot 10^{-3}(x-x_0) - 0.033\cdot 10^{-3}(x-x_0)^2 \\ &\quad +0.166\cdot 10^{-6}(x-x_0)^3 \end{aligned} \tag{110}$$

4.12 Absolute value and position of deviation maximum

For the calculation of the correct position of the deviation maximim of $S(x)$ and $G(x)$, the equation (101) for x_{max} has to be adapted, because the case $d = 0$ is not admissable. The equation is changing to:

$$b + 2c(x_{max} - x_a) = m = \frac{y_2 - y_1}{x_2 - x_1} \tag{111}$$

$$\Leftrightarrow x_{max} = \frac{m - b + 2cx_a}{2c} \underbrace{=}_{m=0} 150 \tag{112}$$

The position with the most deviation is - like assumed - exactly in the middle of $S_1(x)$. The absolute value can be calculated by:

$$S_1(x_{max}) - y_1 = d_{max} = 0.3417 - 0.25 = 0.092 \tag{113}$$

Using this, the maximum deviation of the cubic spline interpolation can be calculated which can be used for predictions about the interpolation quality.

5 Automated documentation of dynamic accuracy

5.1 Automated program for documentation of robot analyses

Automated Program for Documentation of Robot analyses - AProDoR - is a MFC-based Microsoft Windows application [18] for analyzing and documentation of robot measurements. It was developed during this work in collaboration with a german car manufacturer.

5.1.1 Intention of AProDoR

AProDoR was in the first step basically developed to relieve the user in documentation of measured industrial robots. After the measurement of up to 200 different movements, consisting of up to 5000 measurements each, the effort for an objective documentation and analyzation of the data is high, caused if no automated processing of the data is achieved.

For complex calculations, special techniques in cutting one measurement of a single movement trajectory into slices is needed. For the user, this is an exhausting and difficult task because of searching key positions inside of a point cloud of several thousand positions.

AProDoR was meant to solve this problem and to change this time-consuming and error-prone task into an easy-to-use and optimized solution. Based on intelligent routines for finding the key positions inside of the data crowds, the analysis of the robots TCP movements can be done in an objective way.

Added by illustrations of the analyzation results, the generated output by AProDoR can directly be used as presentation material for a quick overview and comparison about the robots dynamic behavior and accuracy.

5.1.2 Functional structure

AProDoR is devided into four different parts, which are each supporting an analyzation of a special movement geometry. Various interpretation and calculation methods are used, to identify the needed parameters for predictions about the resulting robots movement accuracy and behavior.

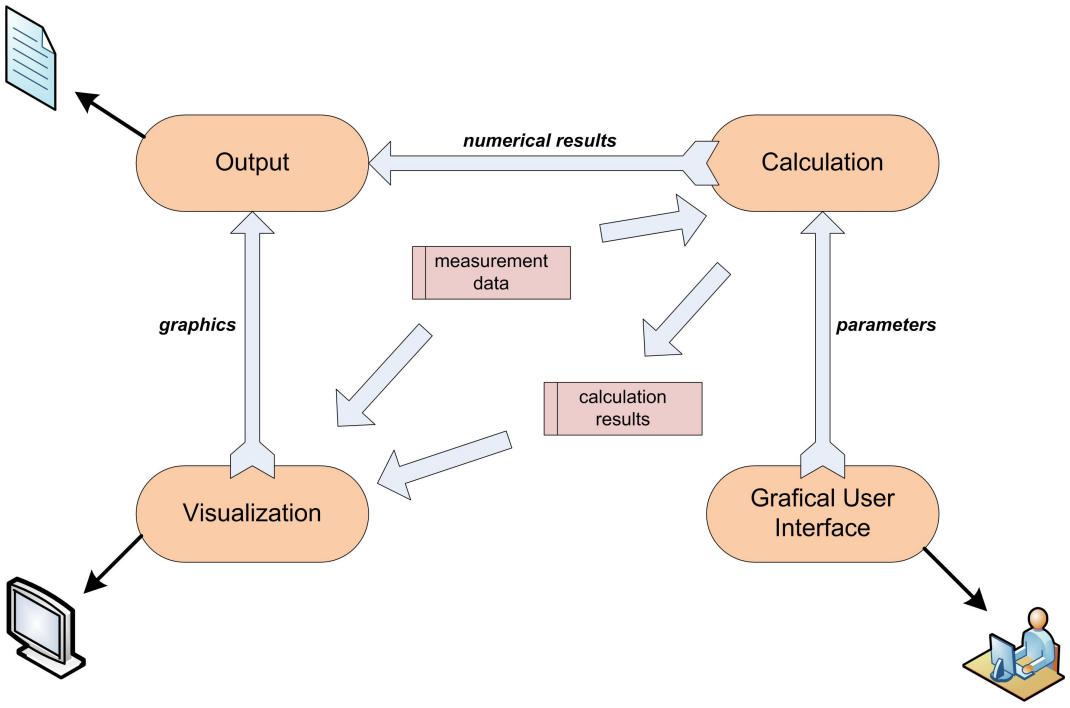

Figure 62: *AProDoR information flow*

The basis structure [17] of all modules, integrated into AProDoR is shown in figure 62. Based on a given set of measurement data, a calculation function is responsible for creating the needed numerical results, like maximum deviation, repeating accuracy or anything else defined by the user (see table 5).

Via a graphical user interface, adapted to each module, the user is able to enter specific data and the corresponding measurement files for one movement analyzation. Different settings concerning number of repetition cycles or used robot movement behaviors are processed by the calculation module

for creating the result output. Also calculation results needed for the visualization module are generated by the calculation module, based on the measurement data.

Velocity or deviation diagrams for example facilitate the online-interpretation of the taken measurement data for the user. The created visualization diagrams are also used for the final documentation output of the software, which is represented by a result-summary in Portable Document File format (PDF).

5.2 Example : Straight line movements

Straight line movements are documented in two parts. For a quick overview about the basic facts of the movement results, an information table like shown in table 5 is generated.

Path length	1.08 m
Ideal velocity	600 $\frac{mm}{s}$
Measured velocity	598.8 $\frac{mm}{s}$
%-velocity deviation (max)	$\pm$ 11.8%
Acceleration	6.6 $\frac{mm}{s^2}$
Max. path deviation	1.29 mm
Repeating uncertainty	0.08 mm
Absolute deviation at start position	1.096 mm
Absolute deviation at end position	0.663 mm

Table 5: *Fact overview in straight line documentation*

It is a summary about the most important analyzation results, delivered by the calculation module (see figure 62).

In the figure 63, the 3-dimensional movement of the robots TCP is split into 3 different plane views in figure 63. Focusing on each 2-dimensional plane, the robots real behavior (shown in red) can be compared to the ideal path (shown in green).

Clearly to see, there are the typical path deviations at the beginning and end of the movement path. This reflects in the diagram for the absolute deviations (figure 64).

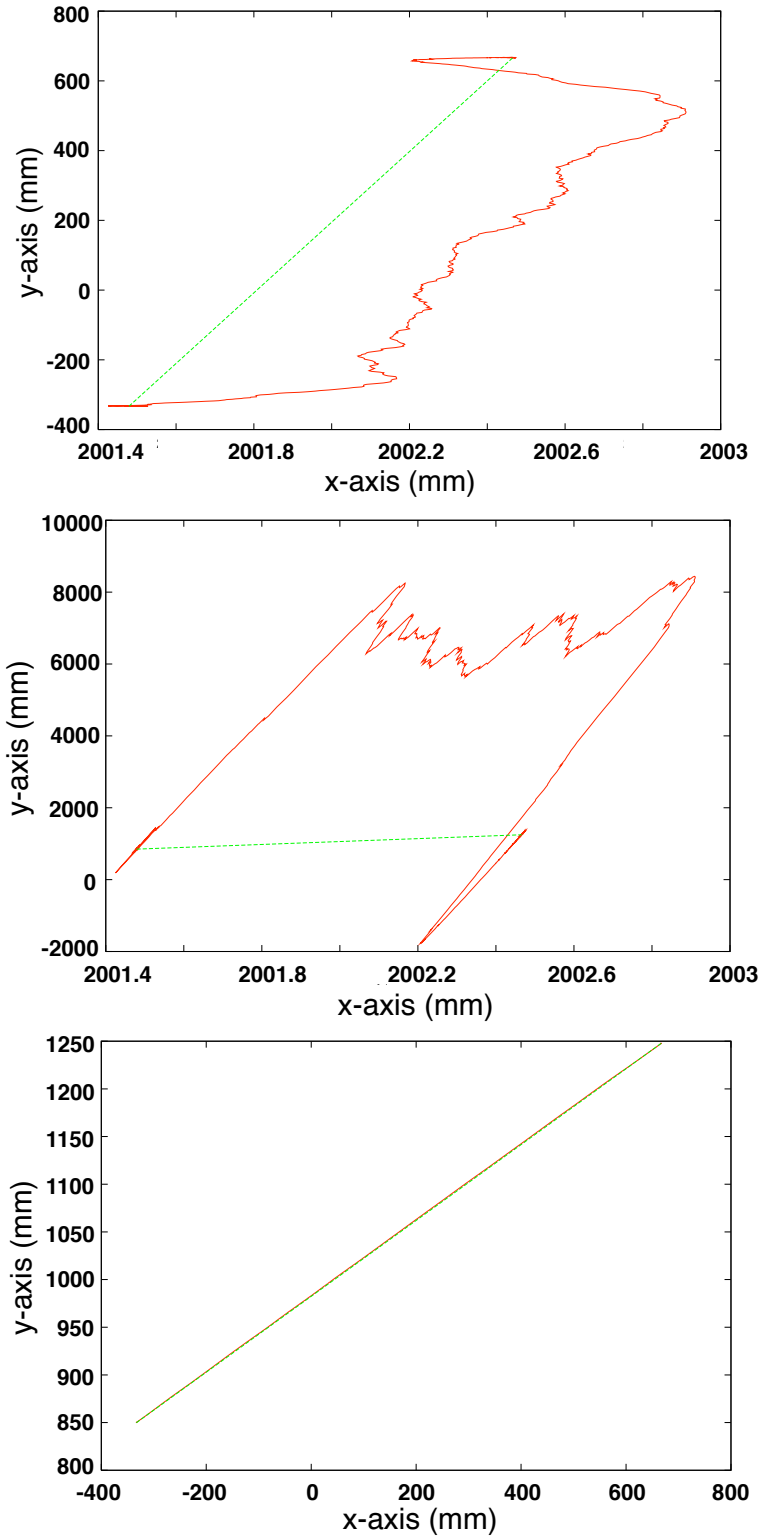

Figure 63: *Different plane views for straight line documentation (green: ideal, red: real)*

The used robot for the application of the presented analyzation was a common industrial robot, normally used for sealing applications in automotive industry. As an example for one of the various opportunities to make predictions about the robots movements behavior, the real measured TCP movement shown in the x/y-plane and x/z-plane diagram of figure 63 is taken.

The red line represents the measured path, which is afflicted with a positional error to the ideal straight line movement. A very important fact to that dynamic error is, that the measured deviation is nearly constant after start acceleration and before end acceleration of the robot arm.

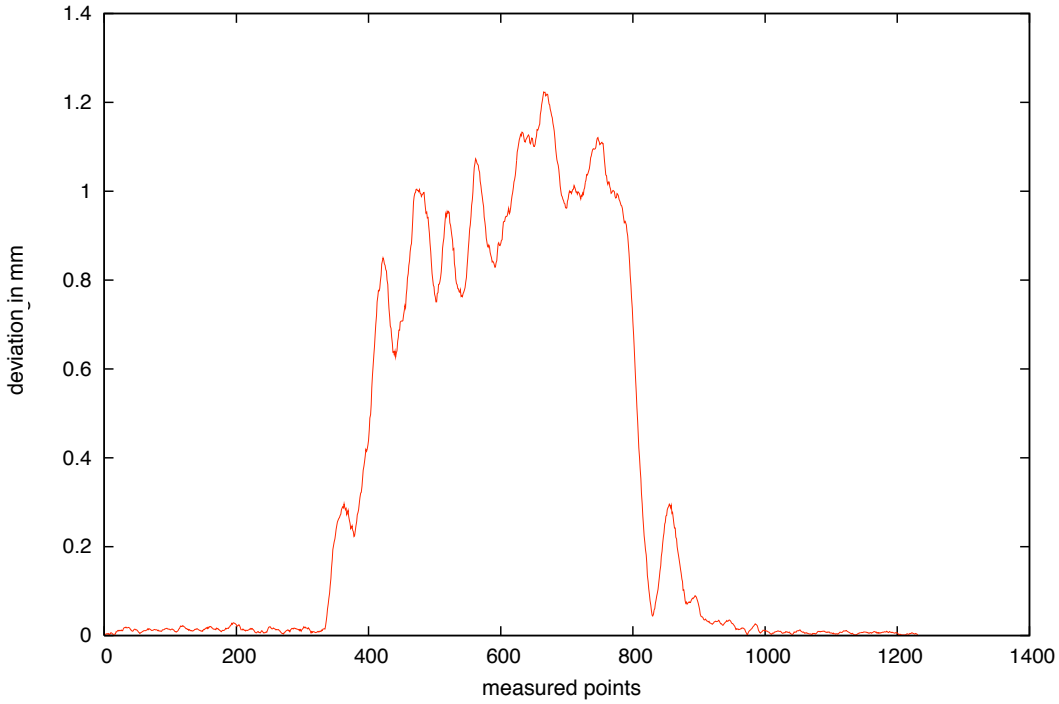

Figure 64: *Absolute deviation of real movement to ideal path in mm*

In combination with the absolute deviation diagram in figure 64, a deviation up to 1.2 mm was calculated (see table 5). This deviation would exceed the tolerated dynamic accuracy in application. But due to the effect, that the real movement is represented like a parallel movement to the ideal line with only $\pm$ 0.25 mm in average, it can be tolerated in application after adjusting the start and end positions.

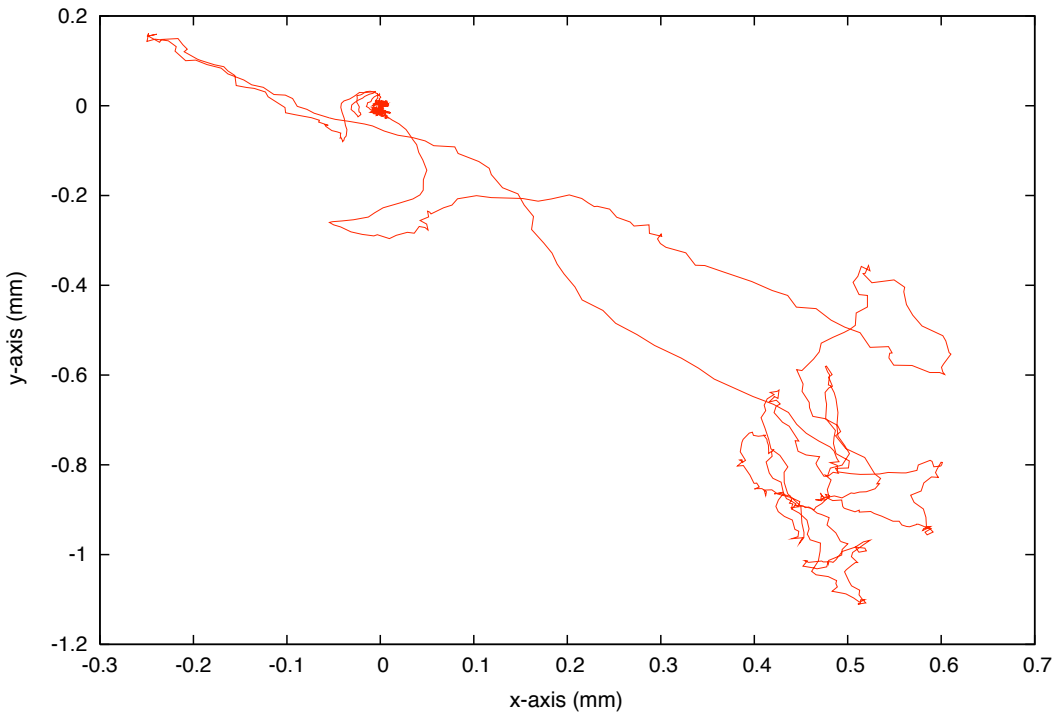

Figure 65: *Look-through diagram of straight line movement*

The parallel movement can also be seen using the look-through diagram (figure 65). Here, even the exact position of the parallel movement line can be located, using the density distribution. A definite zone in the lower right edge with a center-of-gravity at approximately $(0.5\ mm, -0.8\ mm)$ in x/y-plane of the look through can be located.

The dimensions $x = 0.5\ mm$ and $y = -0.8\ mm$ of this area can be estimated to $\pm 0.1\ mm$ in width and $\pm 0.35\ mm$ in height. Corresponding to each vector length of the single points, it is fitting with the prior mentioned deviation zone of $\pm 0.25\ mm$ out of the absolute deviation diagram (figure 64).

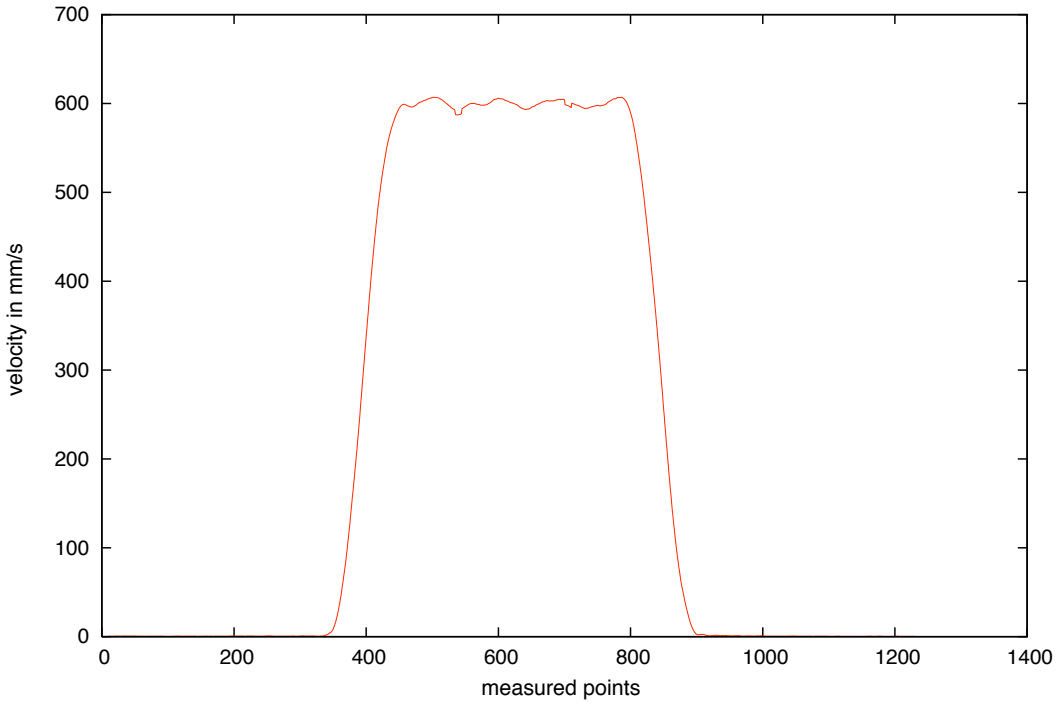

Figure 66: *Velocity diagram*

The velocity for the TCP movement was at a speed of 600 mm/s, which can be seen in the velocity diagram (figure 66) . The resulting acceleration can be calculated from the velocity ramps.

After all, the movement behavior of this robot can be summarized as adequate for applications with needed robot accuracy lower than 0.5 *mm* at movement speeds of approximately 600 mm/s. This example measurements were even taken with a Leica Laser Tracker, which is described more closely in the next chapter.

As a conclusion, AProDoR can be used for rapid-analyzation of robot movements. Within a processing time of several seconds, presentable documentation material is created, only with minimum interaction of the user.

This will be the first step for a high-speed and objective documentation of robot accuracy used for the extremely difficult process of choosing adequate industrial robots for a given application.

6 Uncertainty of the measurement system

All used measurements in this thesis were taken with one special 3D-measurement device. This chapter is dealing with the presentation of the technical features of this system, followed by a theoretical uncertainty calculation.

For expression of the resulted uncertainties, processed by the algorithms and techniques mentioned in this thesis, the so-called combined uncertainty of the measurement has to be calculated. This is done on basis of the description of the International Organization for Standardization (ISO) [20] and on the guidelines for evaluating and expressing the uncertainty in measurements by the National Institute of Standards and Technology (NIST) [19].

6.1 The Leica Laser Tracker

Figure 67: *Leica Laser Tracker LTD800*

During measurements needed for this work, the Leica Laser Tracker LTD800 (figure 67) was used for capturing all 3D-coordinates. It is one of the most accurate and flexible mobile laser tracker in the world today [22]. Using special reflectors, this tracker is able to take non-contact measurements even on moving objects.

Figure 68: *Corner Cube Reflector*

The used reflectors (figure 68) are centrically mirroring the laser beam of the tracker independent to its input angle. Using this, the laser tracker recognizes, if the reflectors position is changed and compensates by adapting the output beam angle on the measurement system.

The measurement system consists of the sensor itself and an additional PC-based sensor controller. The sensor controller is responsible for data processing and communication with external peripherals like a connected user PC for measurement.

Including to the measurement systems there is a programming interface provided, for automation of the measurement process. Using this interface, all software and hardware functions can be executed remotely.

A very important feature of the LTD800 concerning accuracy measurements for industrial robots, is given by its external trigger interface. Over a common RS232 interface, it is possible to used external TTL-based trigger signals for starting and finishing measurements. By this ability, it is possible to synchronize the measurement clock of the laser tracker to every given cycle clock, which is adapted to the trigger interface.

6.1.1 Measurement uncertainty

The combined measurement uncertainty u_c of the Leica Laser Tracker can be calculated for two different cases, depending on the focused measurement space. In case of using a cylindrical measurement shape, the combined uncertainty is expressed in relation to the used measurement radius and the corresponding height in cylindrical coordinates (see figure 69 left side).

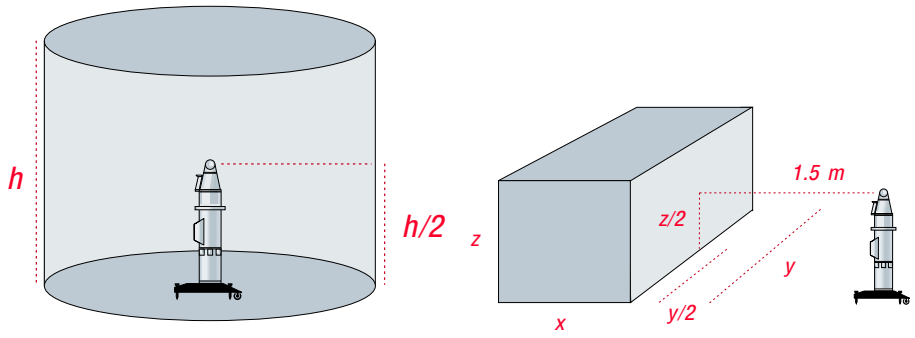

Figure 69: *Uncertainty areas of Leica Laser Tracker*

A common application case for the laser tracker will be to measure a 3D-coordinate. Relating to the x, y, z-representation, the corresponding spanned measurement space will be a rectangular shape. For this, the uncertainty has to be adapted. Both uncertainties, for cylindrical and rectangular case are listed in table 6:

r(m)	h(m)	U_c(mm)
2	2	± 0.055
5	5	± 0.085
10	10	± 0.156
20	20	± 0.297

x (m)	y (m)	z (m)	U_c (mm)
1	1	1	± 0.051
1	2	1	± 0.053
2	2	2	± 0.067
3	3	3	± 0.083
3	6	3	± 0.092
3	10	3	± 0.110

Table 6: *LTD800 combined uncertainty depending on measurement space taken from [23]*

6.2 Combined uncertainty of laser tracker measurements

As a goal for this part of the thesis, is to find out the exactness of the measurements taken by the Leica Laser Tracker. Like every measurement system, even this tracker is afflicted with a dedicated uncertainty. First explaining the term uncertainty itself, the combined spatial uncertainty of a spatial measurement on a fixed ball bar is calculated.

6.2.1 Uncertainty in general

Taking 3-dimensional measurements using any measurement system, is always combined with estimation of tolerances. Every realized system for getting coordinates in a limited measurement range will also have a limited accuracy for creating the output results.

Finite resolution of angle encoders for example, or limited accuracy in distance measurement are causing a bunch of insecurities to the final resulting coordinate of the measurement. These insecurities are commonly called uncertainties.

A complex measurement system is consisting of multiple parts and subsystems, each one responsible for a dedicated task in finding the final measurement result. Every of this subparts of the measurement device will have got

its own finite accuracy, geometry or even time-invariant behavior. Resulting, every subpart is causing its own structured influence to the final measurement uncertainty.

Combining everything, it can be affirmed, that a single measurement result Y - following defined as measurand - is a function consisting of N different influences X_i, called quantities:

$$Y = f(X_1, X_2, \ldots, X_N) \tag{114}$$

Now based on equation 114, it is important to focus on the fact, that a measurand is never a fixed value, but only an approximation of the real result addicted to the concatenation of the single uncertainties of the system components.

The structure of the quantities can be different and is also depending on the way of their use. Basically, all quantities can be roughly divided into two groups:

- Random effects
- Systematic effects

Systematic effects, are all quantities, which can be associated with a known and possible to model behavior in the system. These quantities are integrated in every measurement and can be reduced by an error compensation. Quantities based on random effects are nearly impossible to model and therefore in most cases not easy to compensate.

Each single component X_i in equation 114 has got an own standard uncertainty [21], expressed by u_i. To get the combined uncertainty of the system, all these uncertainties have to be composed together to build the combined standard uncertainty u_c. This composion is based on the law of propagation of uncertainties (see subchapter 6.2.2).

The combined standard uncertainty [19], [20] is commonly used in giving reporting results. In that case, the estimated measurement value and its corresponding uncertainty together are forming the measurement result. With a confidence of 68% is the measurand - so the result of the measurement - inside the interval defined by estimation and uncertainty:

$$Y = y \pm u_c(y) \tag{115}$$

6.2.2 Law of propagation

With a given measurand Y, defined by a function f which contains all significantly influencing quantities included in the system follows:

$$Y = f(X_1, X_2, \ldots, X_N) \tag{116}$$

This measurand is approximated by an estimated measurand, which is again influenced by an amount of - this time - estimated quantities:

$$y = f(x_1, x_2, \ldots, x_N) \tag{117}$$

To calculate the combined standard uncertainty composed from all given standard uncertainties of the estimated quantites, the law of propagation has to be applied:

$$U_c^2(y) = \sum_{i=1}^{N} \left(\frac{\partial f}{\partial x_i} \right)^2 u^2(x_i) + 2 \sum_{i>j} R_{ij} \left(\frac{\partial f}{\partial x_i} \right) \left(\frac{\partial f}{\partial x_j} \right) u(x_i) u(x_j) \tag{118}$$

The correlation coeffient R_{ij} is assumed to zero, if i and j are not correlated.

6.2.3 Input quantities

For the scale bar measurement using a laser tracker, four different measurements can be focused out. These are causing in four different input estimates into the system, more exactly into equation 114. The combination of this four different input quantities results the combined standard uncertainty of the spatial distance measurement.

A basic property of the input estimates is the fact, if they are correlated of independent quantities. The following estimates are inputs of the measurement uncertainty calculation:

- **Angular measurement**
  The uncertainty of angular measurements is depending on the current angle value of the encoder. For this, the influence on the accuracy in measuring the current angle is differing with each measured position. Resulting the angular measurement uncertainty is an independent input estimate.

- **Interferometric scale**
  The interferometric scale of the distance measurement is depending on the current temperature, air pressure and wavelength conditions. It is assumed, that these parameters are constant during measurement and that the user is not changing the integrated compensation parameters of the system. For this case, the interferometric scale will be input estimate correlated with this values for long time observations.

- **Bird bath distance**
  The so-called bird bath distance is the calibrated, initial offset of the used reflector. Its is a high accurate mounting, in which the reflector has to be put and the measurement system has to be locked on before measurement. The theoretical offset, which is used in the laser tracker software, is even an approximation to its real value and addicted to an uncertainty.

  This uncertainty represents two different input estimates, depending on the way, the measurements are done. In the case of not breaking the laser beam after measurement of one point, the bird bath distance can be assumed as a correlated input estimate. In other case, if the laser beam is broken during measurement, the uncertainty of the set bird bath distance is different for each measurement and therefore and independent input estimate.

6.2.4 Spatial measurements

With a geometric mounting like shown in figure 70, two single 3-dimensional points P_1 and P_2 on a scale bar were measured in space. The shape on the

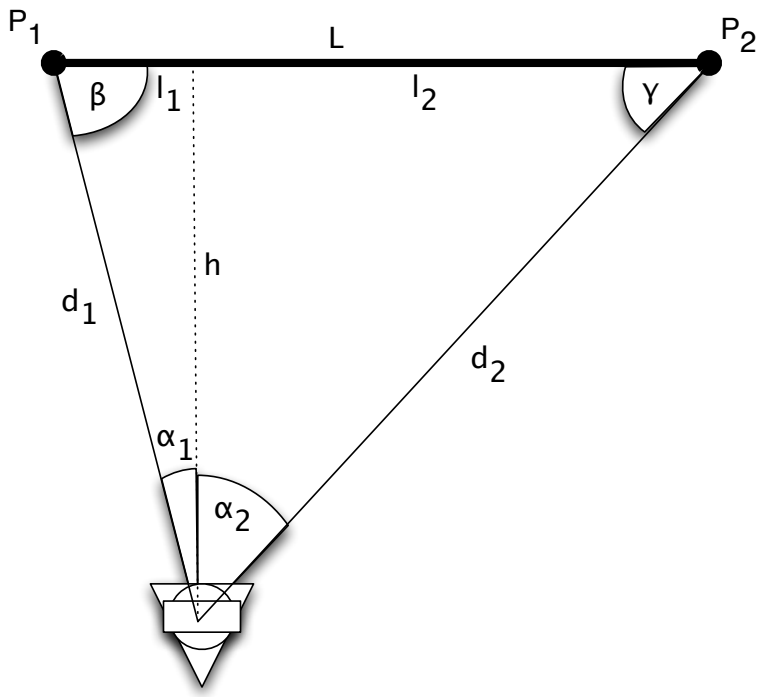

Figure 70: *Spatial measurement of two independent points*

lower end of the depicted triangle represents the laser tracker. This has got an vertical angle α_1 for the measurement of P_1 and an vertical angle α_2 for the measurement of P_2.

6.2.5 Combined standard uncertainty of the used scale bar

The combined uncertainty of the used scale bar is composed out of five different input quantities. Estimates of the uncertainty can be expressed as:

$$y = L(\alpha_1, \alpha_2, m, d_{01}, d_{02}) \tag{119}$$

with

- α_1 : Angular measurement in P_1
- α_2 : Angular measurement in P_2
- m : Interferometric scale
- d_{01} : Birdbath distance P_1
- d_{02} : Birdbath distance P_2

To describe the length of the scale bar, referring to figure 70, the emerged triangle can be split into two rectangular triangles with adjacent use of trigonometric functions:

$$L = \{(d_{01} + m \cdot d_1) \sin \alpha_1 + (d_{02} + m \cdot d_2) \sin \alpha_2\} \tag{120}$$

As a simplification for the following derivations, the initial distances d_{01} and d_{02} are set to zero, based on their small value compared to the point distances. Also, the interferometric scale is assumed as 1, while the small deviations from this value can be neglected.

$$d_{01}, d_{02} << d_1, d_2 \qquad m \approx 1 \tag{121}$$

According to equation 118, the partial derivation of each input quantity has to be constructed. Starting with the partial derivates for the measurement angles follows:

$$\frac{\partial L}{\partial \alpha_1} = \frac{\partial}{\partial \alpha_1} \{(0 + 1 \cdot d_1) \sin \alpha_1 + (0 + 1 \cdot d_2) \sin \alpha_2\} = h \tag{122}$$

In the same way, the partial derivate of L to m can be calculated:

$$\frac{\partial L}{\partial m} = d_1 \sin\alpha_1 + d_2 \sin\alpha_2 = l_1 + l_2 = L \tag{123}$$

More complex is the calculation of the partial derivates for the quantities d_{01} and d_{02}. Here, a case differentiation is necessary, to distinguish between a broken or non-broken beam during measurement.

Case 1 : No beam break during measurement,
d_{01} and d_{02} correlated (Substitution: $d_{01} = d_{02} \rightarrow d_0$):

$$\frac{\partial L}{\partial d_0} = \sin\alpha_1 + \sin\alpha_2 = \frac{l_1}{d_1} + \frac{l_2}{d_2} \tag{124}$$

Case 2 : Beam break during measurement,
d_{01} and d_{02} independent ($d_{01} \neq d_{02}$) :

$$\frac{\partial L}{\partial d_{01}} = \sin\alpha_1 = \frac{l_1}{d_{01}} \tag{125}$$

$$\frac{\partial L}{\partial d_{02}} = \sin\alpha_2 = \frac{l_2}{d_{02}} \tag{126}$$

6.2.6 Combined variance of angle measurement

The quantities d_{01}, d_{02} and m are directly related to the measurement system components. For this reason, the corresponding variances $u_c(d_{01}), u_c(d_{02})$ and $u_c(m)$ are given and constant.

Different to this, the combined variances $u_c(\alpha_1)$ and $u_c(\alpha_2)$ are depending on the angle value and changing for each measurement. The variance for horizontal angle measurements $u_c(H)$ is perpendicular to the variance for vertical angle measurements $u_c(V)$. Resulting a concentration ellipse can be spanned:

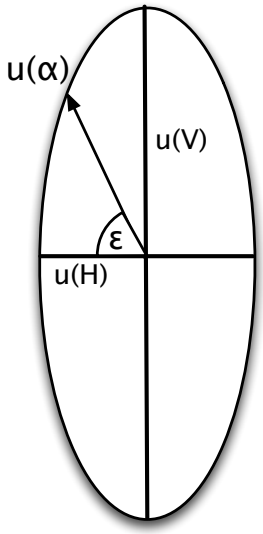

Figure 71: *Error ellipsis of horizontal and vertical measurement*

The error ellipsis, shown in figure 71 is constructed on the bivariate distribution of the measurand, caused by the two uncertainties $u(V)$ and $u(H)$. The measured values are represented in a χ^2-distribution.

The corresponding ellipsis equation can be used to express the structure:

$$r^2 = (a\cos\epsilon)^2 + (b\sin\epsilon)^2 \tag{127}$$

$$\Rightarrow u(\alpha)^2 = (u(H)\cos\epsilon)^2 + (u(V)\sin\epsilon)^2 \tag{128}$$

Corresponding to that, the combined variances u_c^2 for the two independent angle measurements can be calulated by:

$$u_c^2(\alpha_1) = \cos^2 \epsilon \cdot u^2(H_1) + \sin^2 \epsilon \cdot u^2(V_1) \tag{129}$$

and

$$u_c^2(\alpha_2) = \cos^2 \epsilon \cdot u^2(H_2) + \sin^2 \epsilon \cdot u^2(V_2) \tag{130}$$

6.2.7 Lateral error influence

The lateral error component of the measurement system is dependent on the measured angle (horizontal and vertical) and on the distance of the measured position to the laser tracker. In this case, the laser tracker has got a separate specification for near range lateral error:

$$E_{lateral\_H} = c_H, \qquad E_{lateral\_V} = c_V \tag{131}$$

The constants c_H and c_V are used, if a measurement is situated inside of the defined near range of the tracker.

In normal case, if none of the two measured points is inside the near range, the lateral error can be expressed as:

$$E_{lateral\_H} = u(H) \cdot d, \qquad E_{lateral\_V} = u(V) \cdot d \tag{132}$$

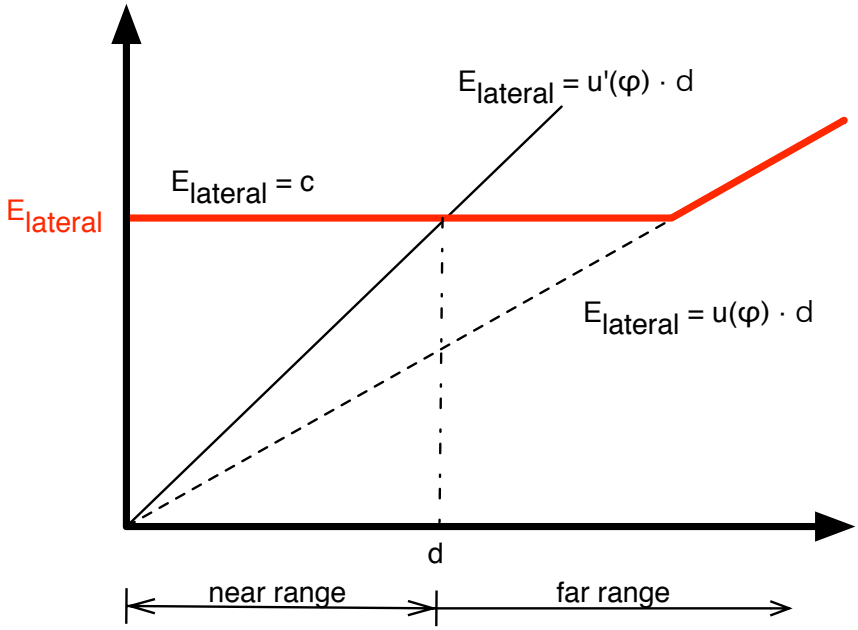

Figure 72: *Substitution graph for far and near range*

The final combined variance $u_c^2(\alpha_i)$ for an angle measurement in the near range now transforms to:

$$u_c^2(\alpha_i) = \cos^2 \epsilon \cdot u'^2(H_i) + \sin^2 \epsilon \cdot u'^2(V_i) \tag{133}$$

6.2.8 Combined uncertainty for spatial distance

Putting everything together, the resulting variance of the spatial measurement in case of no beam break during measurement, is constructed by:

$$u_c^2(L) = h^2 \cdot u_c^2(\alpha_1) + h^2 \cdot u_c^2(\alpha_2) + \left(\frac{l_1}{d1}\right)^2 + \left(\frac{l_2}{d2}\right)^2 \cdot u^2(d_0) + L_2 \cdot u^2(m) \quad (134)$$

In assumption of a beam break, the two uncertainty components $u(d_{01})$ and $u(d_{02})$ can be combined to $u(d_0)$:

$$u_c^2(L) = h^2 \cdot u_c^2(\alpha_1) + h^2 \cdot u_c^2(\alpha_2) + \left[\left(\frac{l_1}{d1}\right)^2 + \left(\frac{l_2}{d2}\right)^2\right] \cdot u^2(d_0) + L_2 \cdot u^2(m) \quad (135)$$

Transforming from variance to combined uncertainty $u_c(L)$ for spatial distance delivers:

$$u_c(L) = \sqrt{u_c^2(L)} \quad (136)$$

7 Coordinate frames and error influences

Today robot application cells are equipped with a high degree of automation. In many high accuracy applications, there are besides of the industrial robots many other systems for improving and support. The robots themselves are put on linear tracks for extension of their workspace.

Additionally object recognizing systems, based on certain 1D, 2D- or 3D-sensors, are implemented to adapt the robot movements to always changing work object positions. Online motion controlling systems, developed to guide the robot along defined movement paths, can be attached to the robots tool.

All these systems have got one basic fact in common: integration. To use a system, which is working to guide, control or support the used robots by means of 3D or 6D coordinates, it has to be integrated in the transformation chains of the application.

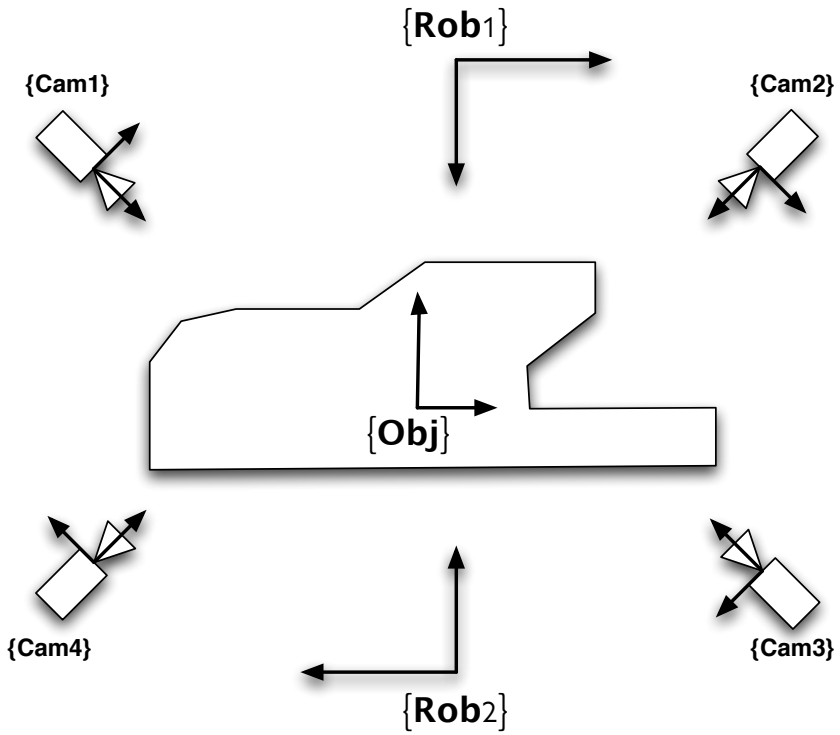

Figure 73: *Standard robot application cell*

To explain the transformation chain, which was focussed in this thesis, figure 73 depicts a standard robot application cell. It consists of two different robots (rob 1 and rob 2), a work object and a 6D vision system (cam 1-4) for object position measurement.

Every component of the application cell has got an own coordinate system, named in the curly braces and denoted by the perpendicular arrows. For an integration of the components, the position and rotation of their coordinate systems must be known in a pre-defined master coordinate system. Figure 74 is showing an example transformation ${}^{master}T_{rob}$ from a master coordinate system to a robot base frame.

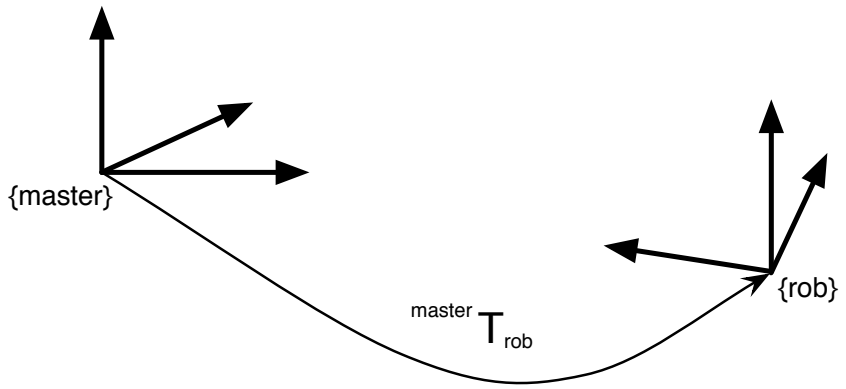

Figure 74: *Transformation from master to robot frame*

Mathematically, this transformation can be expressed as a function depending on six variables, x, y, z for the translation and α, β, γ for the rotation from master to robot.

$$ {}^{master}T_{robot} = f(x, y, z, \alpha, \beta, \gamma) \tag{137} $$

For integration of one component of the robot cell, e.g. the robot itself, the transformation from the master frame to the robots base frame has to be detected. This can be done, based on measurements of several points, which are represented in both coordinate systems.

The equation

$$^{tracker}\vec{T}_{rob} \cdot ^{robot}\vec{p}_n - ^{tracker}\vec{p}_n = 0 \tag{138}$$

can be solved using an iterative method like Newton-Rhapson to find its roots. Using this method, the iterations on the function

$$t(x) = f(x_n) + f'(x_n)(x - x_n) \tag{139}$$

are stopped if the inaccuracy $\epsilon \geq |x_n - x_{n+1}|$.

A basic problem, which is discussed in this subchapter, is the robot accuracy influence on the exactness of transformation calculation. As shown in chapter 3.2, the limited static positional accuracy of the robot is resulting in a positional deflection of its TCP. Furthermore, the saved position coordinates in the robot control are not equal to the real tool positions. The robot is always afflicted with an uncertainty $U(p_n)$ on every coordinate:

$$^{robot}\vec{p}_n = ^{robot}\vec{p'}_n + u(p_n) \tag{140}$$

where the real robot TCP position p_n is consisting of the programmed position p'_n and the uncertainty $u(p_n)$, depending on the single TCP position.

7.1 Effect of point accuracy on robot base coordinate system

As explained before, the appropriate coordinate transformation from the measurement device to the robot base frame, is calculated using two different point sets. The used set of measured point positions generated from the measurement device can be used as fixed values based on the high accuracy of the laser tracker (see subchapter 10).

The other point set, used by the robot for its TCP-positions, is affected with the error components influencing the robots positional accuracy (see subchapter 3.5.2). Using error-affected points for the non-linear least-square-fitting for the transformation calculation, causes a special residual deviation for each coordinate value (figure 75 to 77).

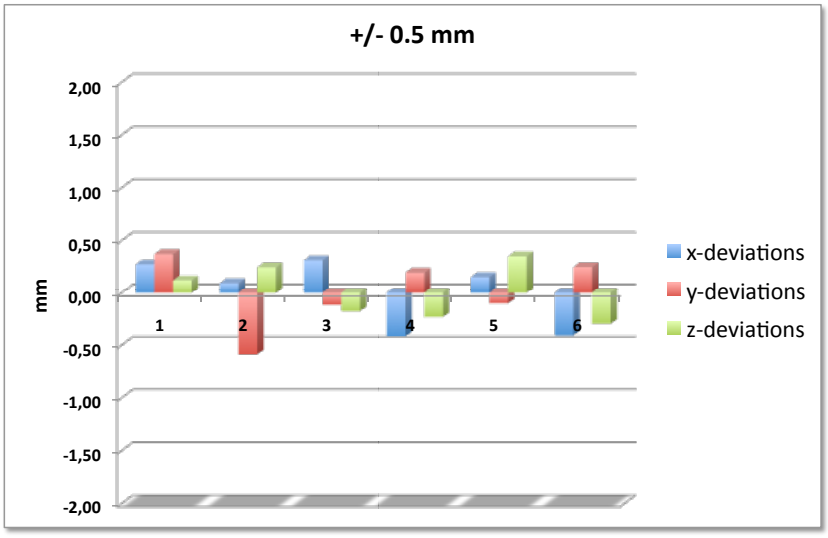

Figure 75: *Residual components using a robot with 0,5 mm accuracy*

Depending on the point accuracy of the robots TCP position, the residual component of every point is changing due to the non-perfect matching of the two differing point sets. Figures 75 to 77 are showing the residual components for three different robot accuracies. As can be seen in the figures, the residuals are increasing with the decreasing of the robots positional accuracy. Even depending on this is the quality of the position and orientation of the

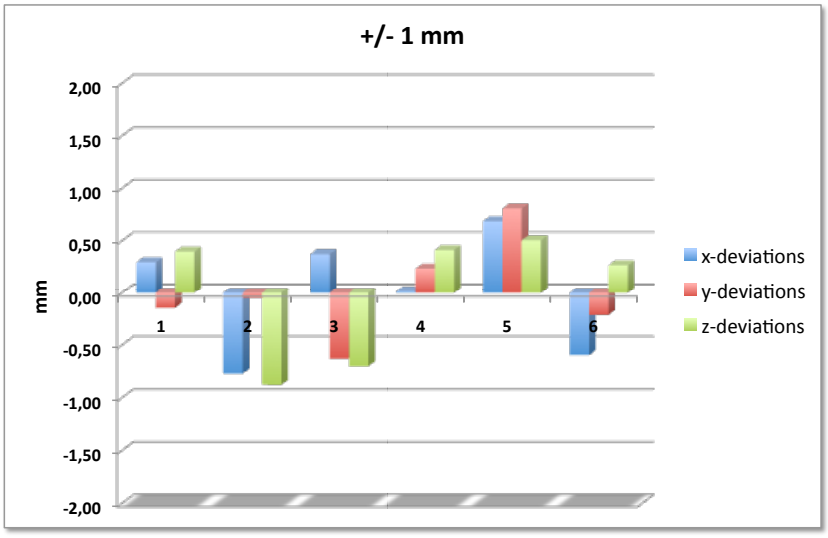

Figure 76: *Residual components using a robot with 1 mm accuracy*

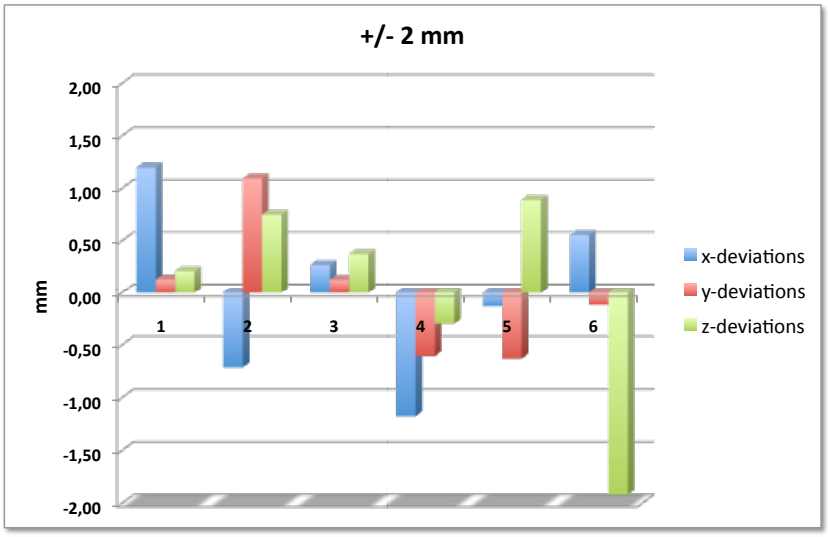

Figure 77: *Residual components using a robot with 2 mm accuracy*

determined robot coordinate system.

More crucial for the integration of the robots into the application by identification of the robot base frame, is the frame error itself. Not only the single point in the sets are transformed with a certain inaccuracy, even the identified base frame is afflicted with an error, depending on the exactness of the used points in the least-squares-fitting.

The resulting base frame inaccuracy is reflected by a 6-dimensional deviation after fitting. Corresponding to the figures 75 to 77, the deviation of the identified base frame is as following:

Point Accuracy	ΔX*[mm]*	ΔY*[mm]*	ΔZ*[mm]*	$\Delta\alpha[^\circ]$	$\Delta\beta[^\circ]$	$\Delta\gamma[^\circ]$
$\pm$0.5 mm	0.07	-0.13	-0.15	359.99	0	0.01
$\pm$1 mm	-0.03	0.15	-0.47	0.02	0.02	0.01
$\pm$2 mm	-0.68	-0.23	2	360	359.98	359.94

Table 7: *Base frame identification deviations*

7.2 Linear track influence on robot base frame identification

As shown before in section 7.1, the position and orientation of the robot base frame is changing, according to the exactness of the used TCP-positions. This is resulting in a permanent deviation between mathematical model of the robot application cell and its real behavior.

An increasing of the robots absolute accuracy will enhance the process of identification of its base frame. To improve the identification process significantly, the robots TCP position has to be measured, using a more exactly method than using its internal sensors.

The identified robot base frame deviations, shown in table 7 are common results based on the identification of more than 100 different industrial robots with handling weights between 16 kg and 200 kg.

Another important fact in the field of identification of the robots base frame are the external influences on the robot during the identification process. The absolute positional accuracy of the robot represents the most crucial error component in the system, but also the external influences are taking a significant part of the resulting inexactness of the robot base frame. Even using a highly accurate robot, the external influence parts are getting more important.

Referred to subchapter 4.2, the most important external influence of the robot is the linear track. Its non-linear behavior is causing an additional deviation part to the robots TCP. The occurring robot twist results in an incorrectly identified robot base frame. Figure 78 is showing a non-linear track with a determined real robot base frame and its theoretical position.

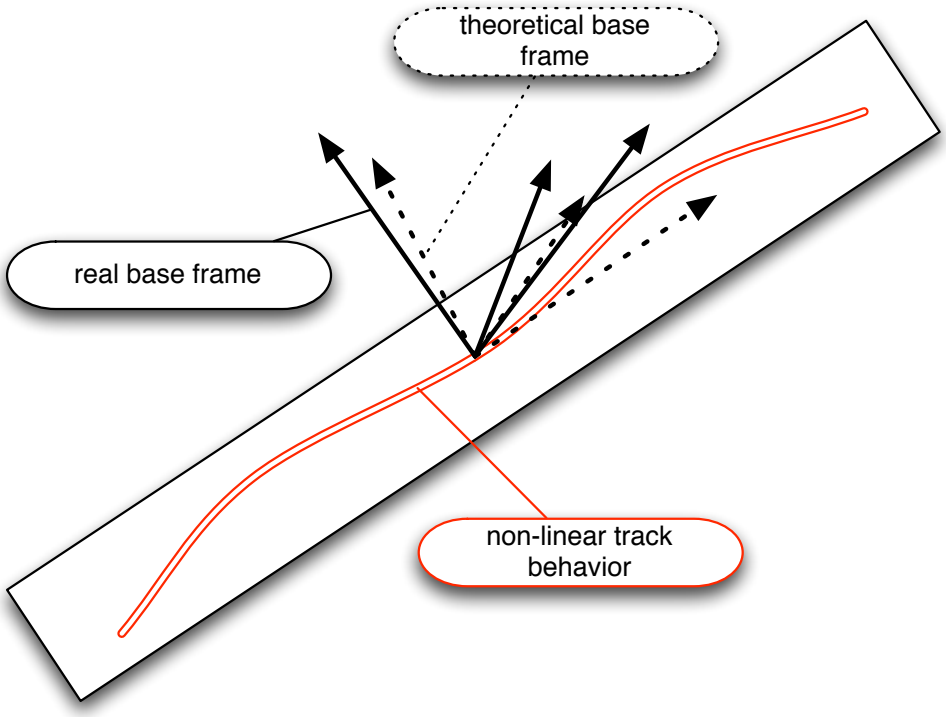

Figure 78: *Base frame shift by linear track*

7.3 Minimization of external influence on robot base identification

One goal during the identification process of the robots base frame is the minimization of the external error influences. To start with the most crucial external influence, the non-linear behavior of the linear track was analyzed using the method described in subchapter 4.3.

Referring to subchapter 10, the accuracy of the measurement system is multiple times better than the identified accuracies in the robot system. With this the measurement error ${}^{meas}\vec{p}_{\Delta,n}$ for each measurement position can be disregarded.

Based on a given real linear track, an identification of the track structure like presented in subchapter 4.3 is done. Figure 79 is showing a non-linear track and two of its measured 6D sampling positions, which are represented by the coordinate systems $\mathbf{T}_n$ and $\mathbf{T}_{N+1}$:

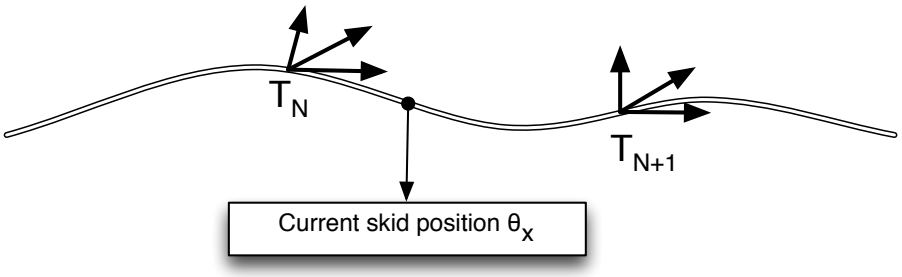

Figure 79: *Point correction after track analysis*

To enable a non-linear compensation of the robot skid positions on the track, a correction value for each possible skid position on the track has to be found. For this, a 6D correction transformation $\mathbf{T}_{corr,\theta_x}$ for each skid position θ_x has to be applied during robot base frame identification measurement.

Following, a complete solution for TCP correction, divided into 4 different steps (see figure 80) is presented.

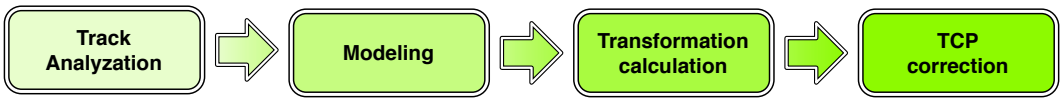

Figure 80: *Solutions steps for TCP correction*

7.3.1 Solution step 1 - Track analyzation

Referring to subchapter 4.3 the linear track is measured in equidistant sampling positions. Each sampling position reflects the 6-dimensional orientation of the linear track and is linked to a determined skid position θ_n. In each sampling position a coordinate system is calculated, to facilitate the 6D-representation. The resulting frame set is further used as a description data set for modeling the linear track deformation.

7.3.2 Solution step 2 - Linear track modeling

As the basis for modeling, a set $S_i(x)$ of cubic spline functions is used to represent the geometrical structure of the linear track. In subchapter 4.6.4 the single spline functions were defined in equation 70.

The sub-spline $S(\theta)$ in an intervall I between two consecutive measured sampling positions θ_n and θ_{n+1} is represented by:

$$S(\theta) = a_n + b_n(\theta - \theta_n) + c_i(\theta - \theta_n)^2 + d_i(\theta - \theta_n)^3, \qquad I = [\theta_n; \theta_{n+1}] \quad (141)$$

Following, the needed coefficients a_n to d_n can be calculated, using the rules for construction of cubic spline functions:

$$a_n = t_x(\theta_n) \quad (142)$$
$$h_n = \theta_{n+1} - \theta_n \quad (143)$$

As marginal condition, the bending factors of the start and the end part of the spline function values are set to zero (d represents the distance offset from the last sampling position θ_N to the end of the linear track):

$$S_0''(0) = S_n''(\theta_N + d) \quad \Rightarrow \quad c_0 = c_N = 0 \tag{144}$$

The recursive calculation of the c_n-coefficients is the main part in the calculation process of the spline function set. The following equation is valid for all c_n with $2 \leq n \leq N-2$ [14]:

$$h_{n-1}c_{n-1} + 2(h_{n-1} + h_n)c_n + h_n c_{n+1} = 3\frac{t_x(\theta_{n+1}) - t_x(\theta_n)}{h_n} - 3\frac{t_x(\theta_n) - t_x(\theta_{n-1})}{h_n} \tag{145}$$

Based on this, the corresponding remaining parameter b_n and d_n can be calculated by:

$$b_n = \frac{t_x(\theta_{n+1}) - t_x(\theta_n)}{h_n} - \frac{h_n}{3}(c_{n+1} + 2c_n) \tag{146}$$

$$d_n = \frac{1}{3h_n}(c_{n+1} - c_n) \tag{147}$$

The needed calculation of an arbitrary correction transformation based on the interpolating spline functions is based on the complete set of splines. This step can easily be done during analyzation process of the track with creation of a basic value table of the spline parameters. The calculated spline function set in this section is only one of six used spline function sets for a complete 6D-interpolation and modeling of the linear track geometry.

7.3.3 Solution step 3 - Calculation of $\vec{T}_{corr,\theta_x}$

The result of the solution step 2 is created by the calculation of 6 sets of cubic spline functions. Each set of spline functions is responsible for the geometrical representation of one coordinate $x, y, z, \alpha, \beta, \gamma$:

$$S_{i,n}(\theta_x) = a_n + b_n(\theta_x - \theta_n) + c_n(\theta_x - \theta_n)^2 + d_n(\theta_x - \theta_n)^3 \tag{148}$$

Corresponding to this equation set, the calculation of a correction transformation at any skid position on the linear track can be determined by filing in the variable skid position θ_x into each of the 6 equation sets. The correction frame T_{corr} itself represents a set of offset values for each coordinate of the current skid position:

$$T_{corr}(\theta_x) = f(\Delta t_x(\theta_x), \Delta t_y(\theta_x), \Delta t_z(\theta_x), \Delta\alpha(\theta_x), \Delta\beta(\theta_x), \Delta\gamma(\theta_x) \tag{149}$$

7.3.4 Solution step 4 - Correction of current TCP position before measurement

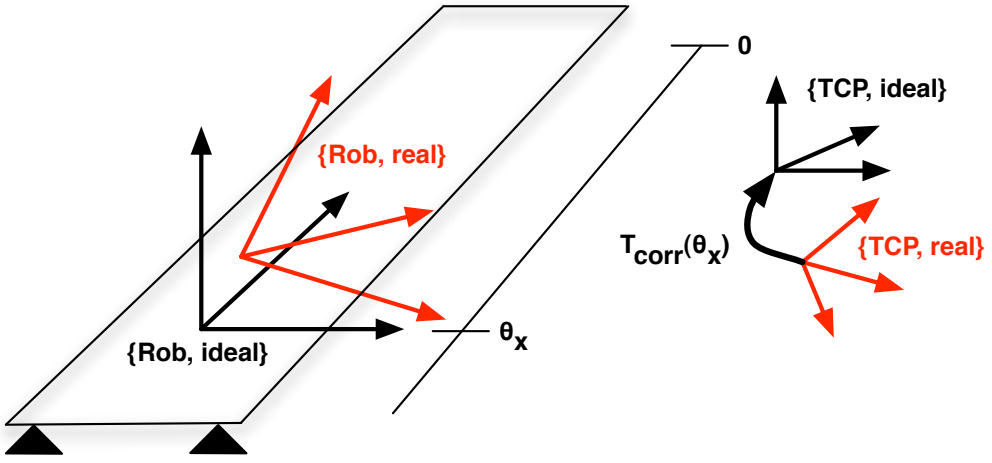

Figure 81: *Correction frame for skid position θ_x*

Figure 81 is showing the effect of the correction transformation on the robots tool center point (TCP), depending on one fixed skid position θ_x on the track.

The final implementation of the correction frame $T_{corr}(\theta_x)$ into the robot application can be done by modification of the current robot work object frame. With a given transformation from the robots ideal position to the object ${}^{rob,ideal}T_{object}$, the corrected robot to object transformation ${}^{rob,real}T_{object}$ can be calculated by:

$$ {}^{rob,real}T_{object} = {}^{rob,real}T_{rob,ideal} \cdot {}^{rob,ideal}T_{object} = T_{corr} \cdot {}^{rob,ideal}T_{object} \qquad (150) $$

Additionally, the ideal position of the robot is depending of the current linear track offset to the zero position of the robots base frame, thus:

$$ {}^{rob,real}T_{object} = T_{corr}(\theta_x) \cdot T_{RobBase} \cdot T_{trans}(\theta_x) \qquad (151) $$

The application of the correction transformation for each of the 6D-coordinates in an arbitrary fix position of the robot skid on the track, causes an adaption to the theoretical model of the application cell and with that, a decreasing of the identification residuals.

7.3.5 Results

As an example for the effect of linear track error compensation on the identified robot base frame, a common industrial robot was measured two times. The robot is a high-accurate robot, used on a significantly deformed linear track. The reference points for the identification are used as repeated robot values, to avoid the static accuracy influence on the measurements. The point residuals for every coordinate and their absolute values are shown in table 8 for both cases, before and after using the linear track correction.

without track correction				
Position	ΔX*[mm]*	ΔY*[mm]*	ΔZ*[mm]*	*rms [mm]*
Point 1	1.772	-1.084	-0.468	2.130
Point 2	-1.091	0.562	0.842	1.489
Point 3	0.834	1.074	0.393	1.415
Point 4	-0.791	-0.571	-1.035	1.422
Point 5	-0.870	-0.910	-0.120	1.265
Point 6	-0.019	1.560	-0.466	1.629
Point 7	0.165	-0.631	0.854	1.075

with track correction				
Position	ΔX*[mm]*	ΔY*[mm]*	ΔZ*[mm]*	*rms [mm]*
Point 1	0.158	-0.058	-0.166	0.237
Point 2	0.100	-0.189	-0.263	0.339
Point 3	-0.361	-0.170	0.044	0.401
Point 4	0.062	0.429	0.030	0.434
Point 5	-0.220	0.097	0.352	0.426
Point 6	0.095	-0.260	-0.002	0.277
Point 7	0.166	0.151	0.006	0.224

Table 8: *Residuals during frame identification*

The resulting root mean square deviation of the used seven positions is also depicted in figure 82. The final accuracy of the measured points, based on the smaller residual values, was improved up to the repeatability accuracy of the robot.
The effect on the identified robot base frame is clearly identifiable. As presented in table 9, the x-coordinate position has a change of nearly 7 mm.

	X[mm]	*Y[mm]*	*Z[mm]*	$\alpha[^\circ]$	$\beta[^\circ]$	$\gamma[^\circ]$
Rob Base before	-750.002	1076.470	-1318.637	358.194	0.799	270.43
Rob Base after	-757.183	1077.449	-1316.749	358.299	1.004	270.753

Table 9: *Identified robot base frame before and after track compensation*

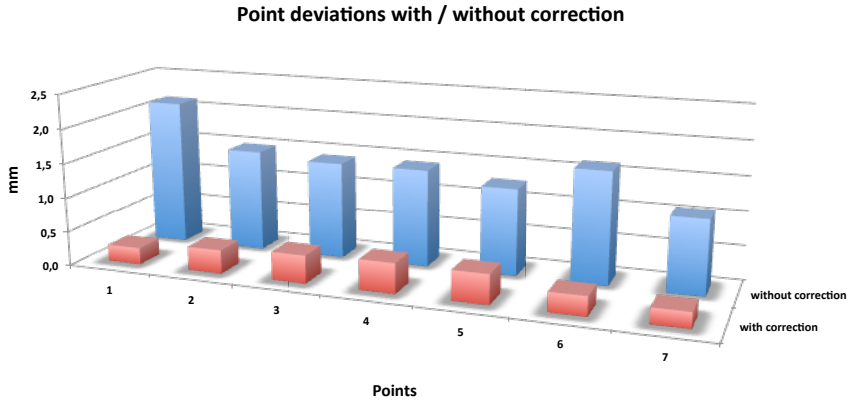

Figure 82: *Point deviations after Least-Squares fitting for robot base frame identification*

In modern application, the positions of the robots base frames in an application cell are very important for the offline-to-online converting process of the robot programs. If the base frames are identified with a positional deflection included, this error is also integrated into every programmed TCP position. For that, the final accuracy of the prepared robot programs is directly depending on the exactness in the identification process of the robot base frame.

8 Copying and mirroring of robot programs

The offline programming of robot application cells in modern industry is focussing on the multiple advantages of CAD simulation. One of this advantages is the development of complex robot movement programs. The process of program construction can be applied during mounting and setup of the application cell.

Nowadays, application cells in automotive industry are structured in more cases in a symmetrical sense, based on the symmetrical geometry of the workpieces they are working on. Robot tasks applied on one side of the workpiece can be adapted to the other side of workpiece by mirroring the movement paths (figure 83).

To increase the working output, multiple robot application cells with same tasks can be put together to a parallel working unit. Each working cell is structured equally and is equipped with the same robots. A robot program, created in one cell can be copied to another cell, to save the effort for recreation of the movement trajectories.

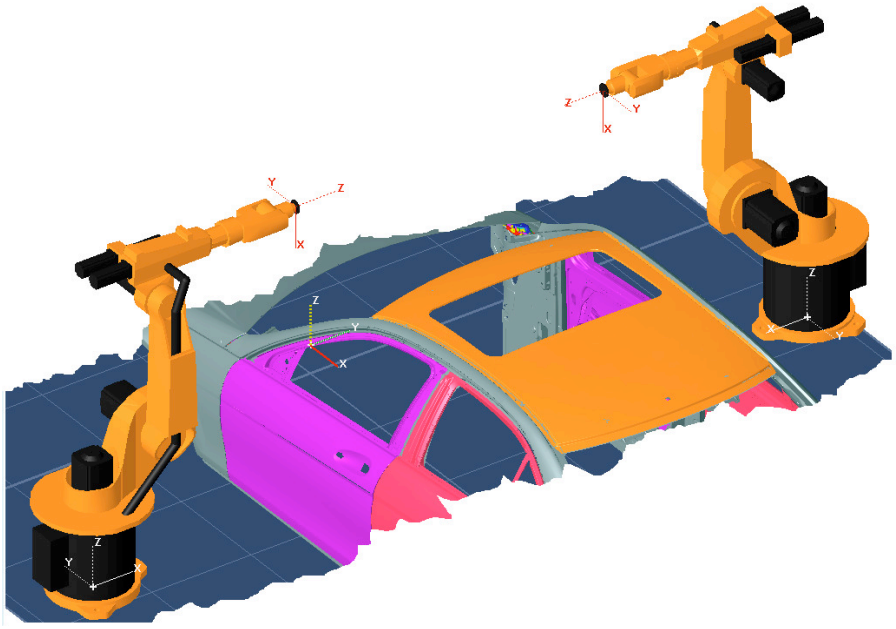

Figure 83: *Symmetrical workpieces enable program mirroring*

The enormous reduction of time and effort for setting up all needed robot programs is increasing the importance of copying and mirroring. In theoretical sense, focussed only to the robot and based on perfect conditions, these actions are easy to apply. Considering practical influences on the robot, caused by its own inaccuracy or the non-linearity of the track, it gets more complicated.

The problems and tasks during the mirroring and copying process of robot programs, especially caused by the linear track is focussed in this chapter. A solution for copying and mirroring robot programs including their track profile is presented below as one of the contributions of this thesis.

8.1 Problems and tasks

Symmetry

The process of copying and mirroring of robot programs is depending on multiple prerequisites. The most important is the symmetry of the application cell. Realizing a robot workflow on one side of a workpiece and transforming it to the other side of the workpiece implicates, that the robot program can be moved, using the same robot movements only mirrored.

More important for the copying process, is the need for equally constructed robot cells. If the different robot cells for which the copying of robot programs is adapted, are designed equally, a re-use of the robot programs created in one cell can be fulfilled much easier in a different cell.

Included into this, the robots should be from the same type and model and mounted against each other. Figure 84 is showing such an example realization. To simplify the mirroring process, the robots base frames should be orientated to have two axes of their coordinate system parallel to the mirror plane. The robots in figure 84 are meeting these prerequisites and their programs are mirrorable.

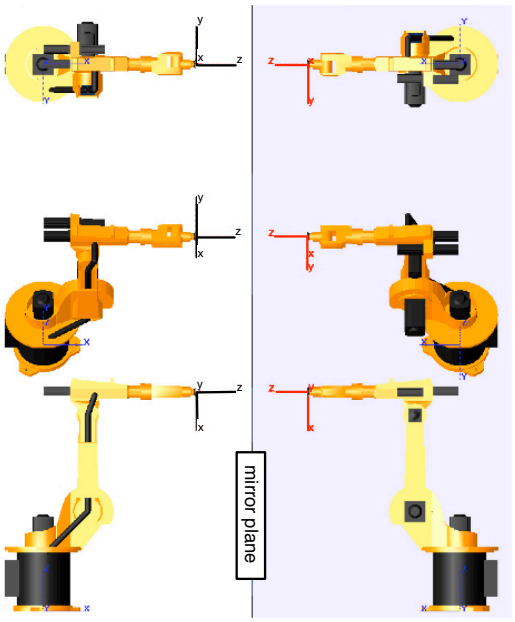

Figure 84: *Symmetrically mounted robots in three different views*

Tool configuration

Several experiments concerning the ability of mirroring modern programs for industrial robots adduced the fact, that even if all needed symmetry aspects of the application cell are considered, in some cases the robot programs are not mirrorable. This is depending on the possible configuration of the robots tool.

Referred to [24] the used robot tool has to fulfill several geometrical restrictions. Two basic parameters are defining the ability to mirror the program: The used mirror plane and the resulting transformation axis to a right-handed coordinate system (see [24] for deeper information). Figure 85 is showing possible transformation axes to get a right-handed coordinate system after mirroring.

In some cases, the used robot tool is equipped with several TCPs. Analyzed tool prototypes for sealing guns were using three different spray nozzles for the sealing process. These were all situated at a different position and - more

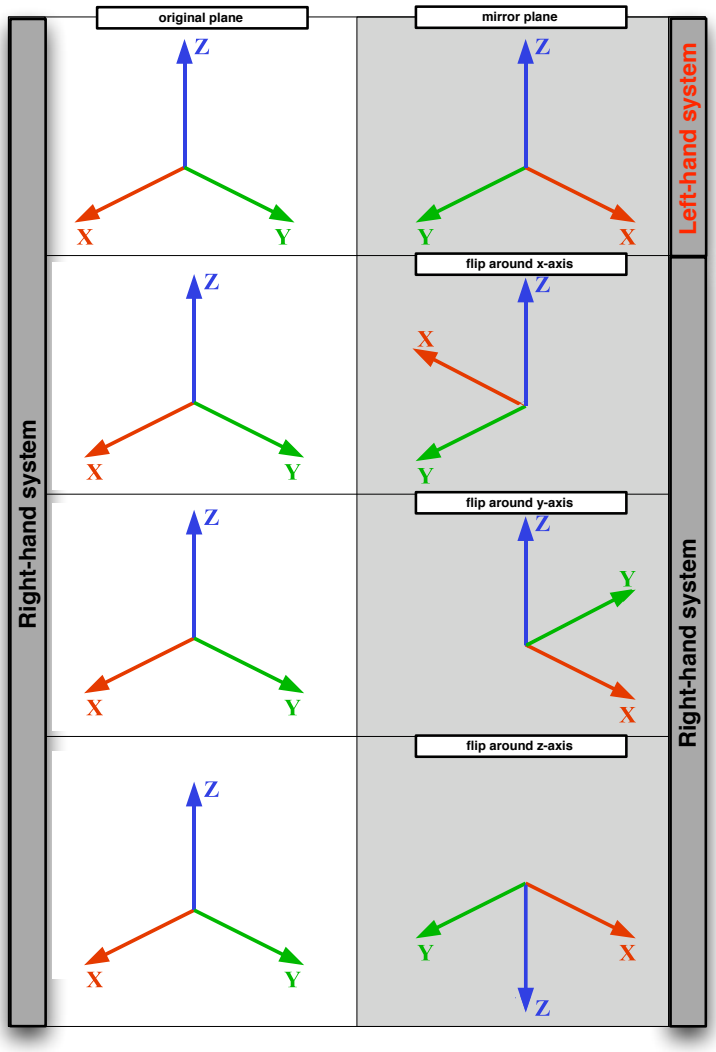

Figure 85: *Possibilities for creation of the right-hand coordinate system after mirroring*

important - at a different angle on the tool. The geometrical constellation of these three TCPs is taking crucial effect on the mirroring process.

Robot accuracy

The accuracy of the robot components, in combination with the accuracy of its sensors is resulting in a positional accuracy of the TCP (subchapter 3.2). In practical application, every robot has got its own accuracy behavior with different deviations in the same TCP positions.

This robot own and unique behavior is causing problems during the mirroring of robot programs. Once corrected a program in online mode, this program will not work with the same point accuracy on another robot. In most cases, the resulting TCP point accuracy after mirroring of online-corrected programs will be even worse than the usage of non-corrected offline programs.

One work-around for this problem is to mirror the generated robot programs in offline-programming phase. For easy transformation, the point coordinates are represented in work object frame. Without improving the robot accuracy to a much higher level, the resulting point accuracy using offline programming will always be afflicted with the robots absolute accuracy.

Linear track

A similar effect can be related to the linear track influence during mirroring. Referring to chapter 4, the linear track is geometrically deformed. This deformation is causing in non-linearities of the track course and with this in inaccuracies of the TCP positions.

The fact, that every linear track is deformed in a different way during the process of assembling, the generated non-linearities will always be different with each different linear track. The important difference to the robot accuracy problem is, that the linear track inaccuracies can be compensated using the correction model presented in this thesis.

For every analyzed linear track, an unique profile has to be processed and a compensation calculation for every program point has to be applied. All based on one linear track - for which the robot programs were corrected online one time - which is dealing as the basic system. A difficult step and a problem, which will be solved in the following section, is how to adapt a linear track profile on online corrected positions, which are already influenced

by a different track profile.

8.2 Transformation calculation

Result of the mirroring process is a modified robot program on a robot B, which is able to work on an online corrected program optimized for robot A. As simplification for this process, some prerequisites have to be considered. For geometrical easyness, the robots should be mountet against to each other in the same working cell. Also the type and model of the robots should be the same. This will have important influence on the robots accuracy, which is involved in the calculation process. Common in practical sense and devolved into the following calculations, the used robot tool is assumed as equal for both robots.

In general, by mirroring of single point coordinates of a robot program, the transformation to a 6-dimensional position has to be mirrored from the robot A to the robot B system. This can be done by using the matrixes s_a and s_b:

$$T_m = s_a \cdot T \cdot s_b \tag{152}$$

This matrixes are represented as diagonal matrixes with +1 and -1 in the diagonal axis, depending on the used mirror plane.

$$s = \begin{pmatrix} \pm 1 & 0 & 0 & 0 \\ 0 & \pm 1 & 0 & 0 \\ 0 & 0 & \pm 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{153}$$

Both matrixes s_a and s_b together are mirroring a coordinate system T to a coordinate frame T_m. The mirror transformation can be split into two separate parts, the pure mirroring and the reconstruction to a right-handed coordinate system.

The first is done by matrix s_a, which is only pure mirroring the system. The resulting coordinate system will be of a left-handed structure. To correct this, the matrix s_b is applied and reconstructs to a right-handed system by flipping one coordinate axis.

The basic element for mirroring a robot program is the transformation to the destination robot TCP in the object coordinate system ${}^{obj}T_{TCP,m}$. As an assumption, both used robot tools have the same geometry, thus the tool transformation for both robots are equal:

$$ {}^{flange}T_{TCP} = {}^{flange,m}T_{TCP,m} \tag{154} $$

For a better overview, figure 86 is showing the complete transformation tree of all considered components in process.

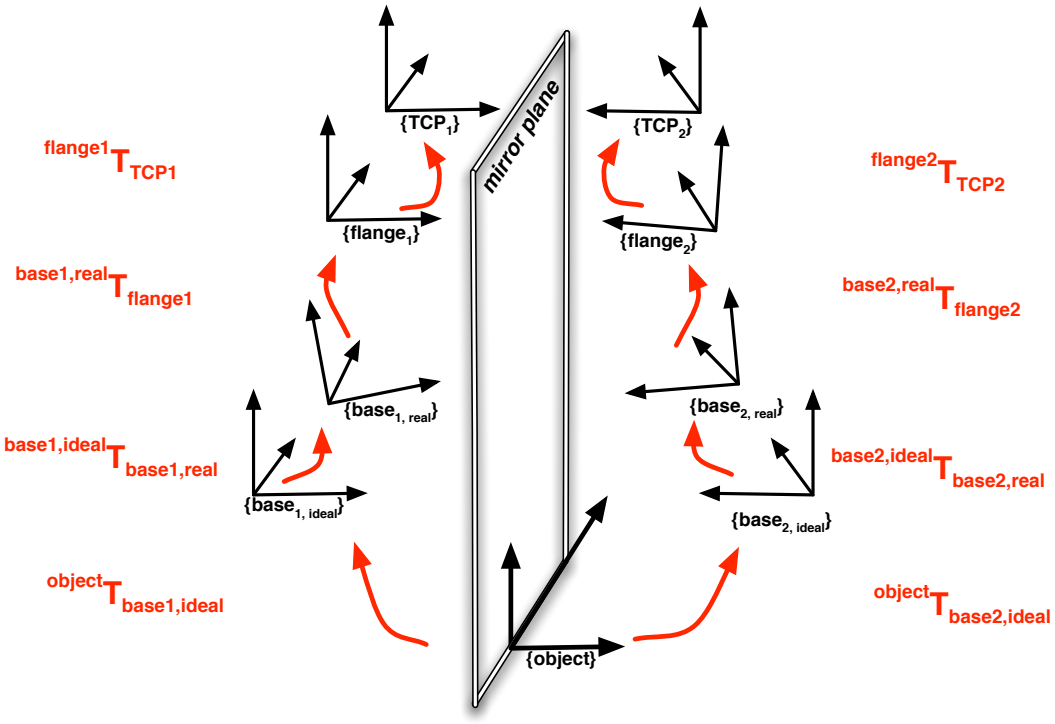

Figure 86: *Transformation graph for program mirroring*

The searched transformation from object to the mirrored TCP can now be expressed as:

$$
\begin{aligned}
{}^{obj}T_{tcp,m} &= {}^{obj}T_{tcp} \cdot {}^{tcp}T_{tcp,s} && (155)\\
&= s_a \cdot {}^{obj}T_{tcp} \cdot s_b && (156)
\end{aligned}
$$

Where s_a is depending on the applied mirror plane and s_b after mirroring for reconstruction to a right-handed system. Including the non-linearity of the track into the system of transformations, the calculation of ${}^{obj}T_{tcp,m}$ changes:

$$
\begin{aligned}
{}^{obj}T_{tcp,m} = && (157)\\
= \; s_a \cdot {}^{obj}T_{base1,ideal} \cdot {}^{base1,ideal}T_{base1,real} \cdot {}^{base1}T_{flange} \cdot {}^{flange}T_{tcp} \cdot s_b && (158)\\
{}^{obj}T_{base2,ideal} \cdot {}^{base2,ideal}T_{base2,real} \cdot {}^{base2}T_{flange,m} \cdot {}^{flange,m}T_{tcp,m} && (159)
\end{aligned}
$$

The following table 10 is showing the applied coordinate transformations and their corresponding accuracy influence to the main transformation from object to TCP.

Mirroring a robot program from one robot to another, implies in many cases same robot models and type on both sides. Based on even equally configured robot controls, the resulting positional accuracies of the robots will nearly be the same. With this, all transformations which are influenced by the robots will have an equal inaccuracy behavior.

Under usage of the same tools on both robots, additional the tool transformation ${}^{flange}T_{tcp}$ from the robots flange to its TCP will be the same for both robots. Differing to that, the correction transformation ${}^{base,ideal}T_{base,real}$ for the linear track influence will differ significantly between the used robots.

${}^{obj}T_{base,ideal}$	:	Including identification error (subchapter 3.5.1), mainly influenced by robot absolute accuracy
${}^{base,ideal}T_{base,real}$	:	Compensation of linear track error influence. Depending on non-linear behavior of the track
${}^{base}T_{flange}$	:	Robots forward-transformation including absolute accuracy
${}^{flange}T_{tcp}$	:	Tool transformation, accuracy depending on identification method for tool data

Table 10: *Used transformation in figure 86 and their error influence*

Even using a same linear track type on both sides, the non-linearities of the two different tracks will be totally different. During construction process, these non-linearities are affected by insufficient measurement and non-perfect positioning of the track rails.

8.3 Transformation of track profile

In case of mirroring a programmed TCP position $P_{x,y,z}$ to another robot without taking consideration to the different track profiles, the final TCP position on the mirrored side will not be reached correctly (see figure 87).

The programmed position $P_{x,y,z}$ will include the track correction $WO(l)$, which will not be the same correction for the mirrored side.

To solve the problem of different linear track profiles on each side of the application and therefore different correction values for the robots in each skid position on the track, first the tracks have to be analyzed (subchapter 4.3). After that, the geometrical structures of the linear tracks can be described like presented in subchapter 4.6.

Each linear track has then to be corrected based on non-corrected data. But, in normal case, one side of the application is already optimized online and manually corrected in every point. It is then not possible to apply the linear track correction like in subchapter 4.7.

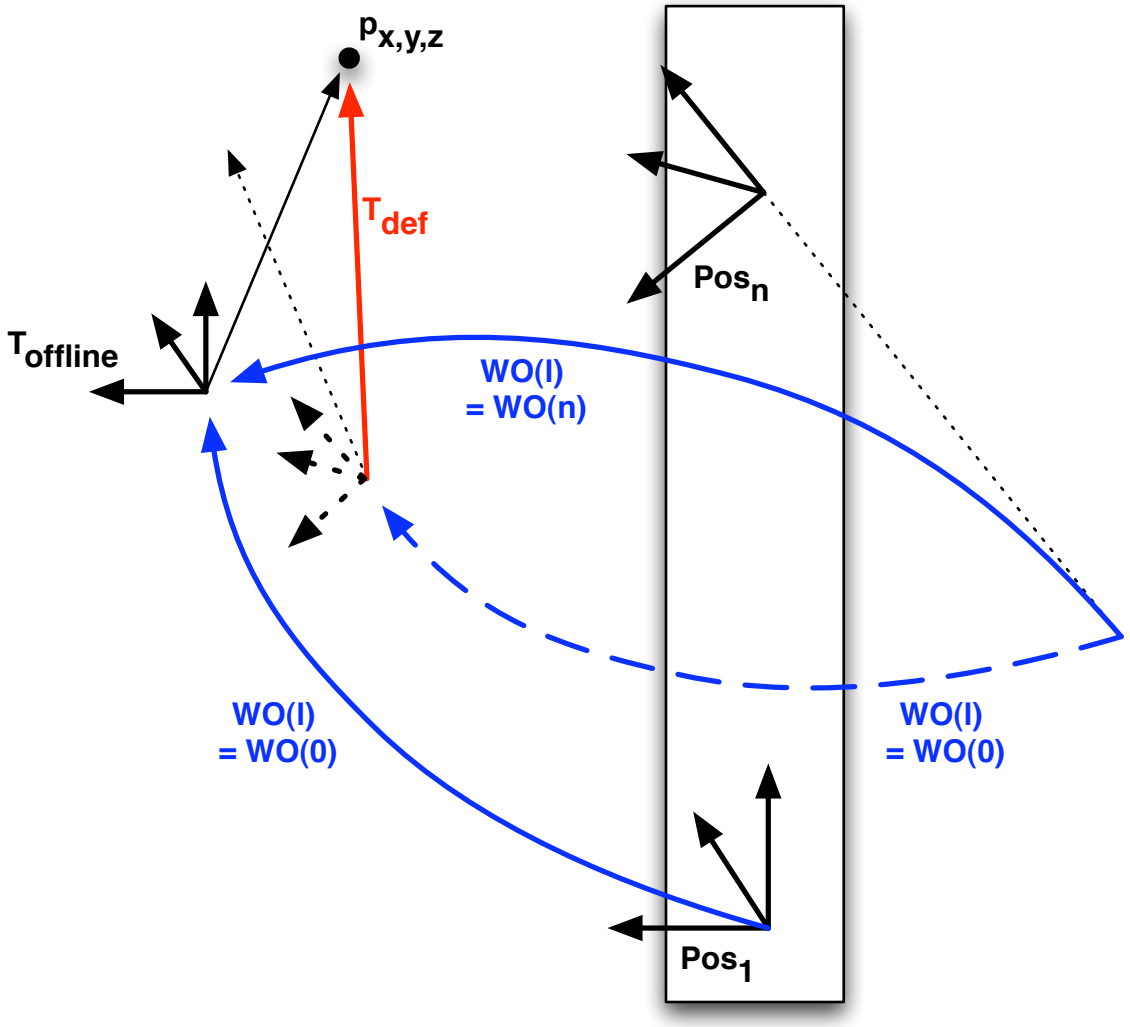

Figure 87: *Linear track profile transversion*

To apply the linear track correction properly after mirroring, the program values have to be calculated to a perfect track profile without any non-linearities. By this, the correction values - applied before mirroring - are compensated and a new track compensation for the other side can be applied. Referring to figure 87, the correction value T_{def} is calculated by:

$$T_{corr} = \{WO_1\}^{-1} \cdot WO_2(l) \cdot T_{offline} \tag{160}$$

and with replacing of $T_{offline}$:

$$T_{corr} = \{WO_1\}^{-1} \cdot WO_2(l) \cdot \{WO_1\}^{-1} \cdot WO(0) \cdot T_{offline} \tag{161}$$

The used transformations $WO_1(l)$ and $WO_2(l)$ are representing the track position depending correction value, with a zero position in $WO(0)$ and a programed TCP position $T_{offline}$.

Using the method of recalculation of an already online-corrected robot program, is becomes now possible to mirror this programs to another robot with an additional application of the track correction. This offers a much better accuracy behavior of the robot programs by reducing the error component in ${}^{base,ideal}T_{base,real}$.

This new method, approached during this work is an increasing of the needed flexibility in the process of robot program construction. Its time and work saving feature helps to decrease the setup cost for robot application cells in modern car industry.

9 Contributions

The set main goal of this thesis, the optimization of robot programs, which were created using offline CAD-systems is reached, based on five different fields. Figure 88 is showing this different contribution fields and their corresponding importance in this thesis. The fields are linked to each other and are all taking influence to the resulting robot application accuracy.

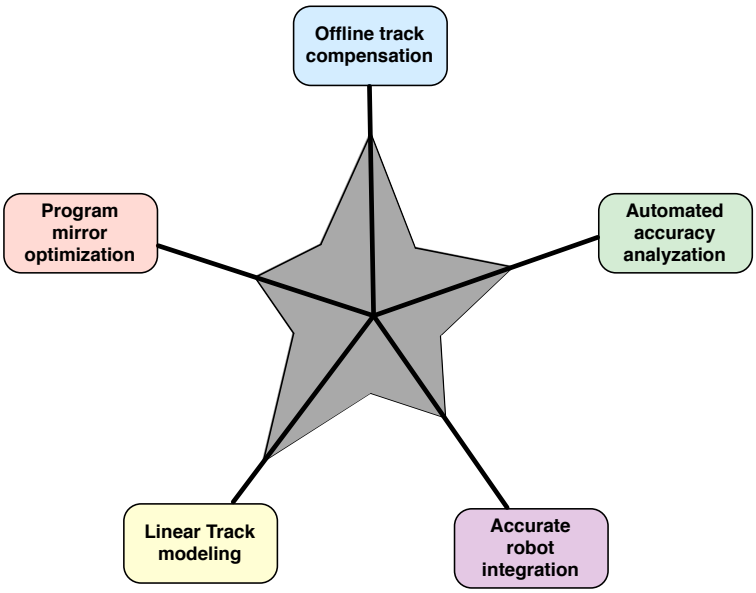

Figure 88: *Contribution fields and importance*

Each of this contributions fields is taking consideration of important steps during the setup of the application. In this thesis, they are composed together to guarantee a maximum level of optimization effectiveness.

9.1 Accurate robot integration

In this work, several parts of the robot system were identified or adjusted, using a high level of exactness by support of the 3D laser tracker (figure 89). I used the identification on the one side and the - if possible - improvement or compensation of the identified robot system parts on the other side for creation of minimum robot accuracy influences for the following steps.

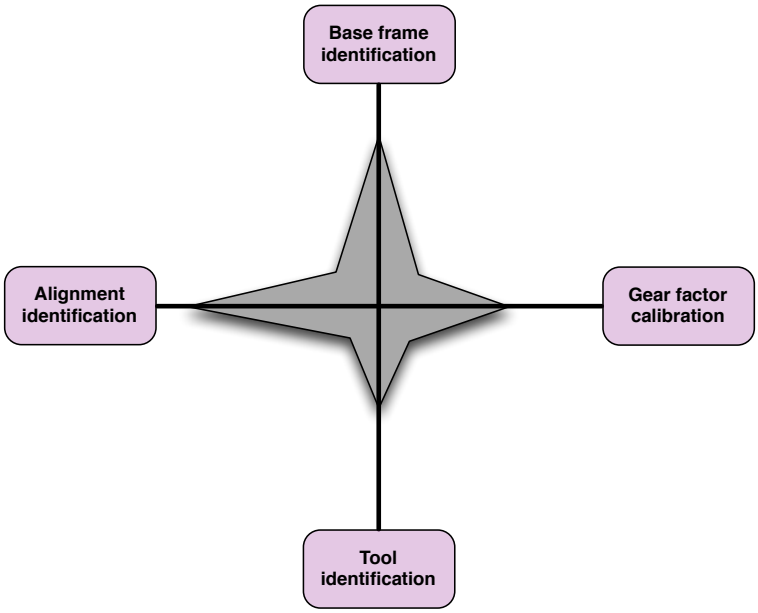

Figure 89: *Contribution facets of robot optimization*

9.1.1 Tracker aided tool identification

A common way to analyse the robots tool transformation is to move the robot to one fixed 3D-coordinate, using different tool orientations, respectively arm configurations. The used fixed position can be a peak, to which the robot tool - in most cases also a mounted peak - is moved to. If the robot tool peak hits the fix position peak, the robot user saves the position of the current tool position for the later calculation by the robot integrated routine.

In subchapter 4.4, I did an improved method of this procedure. Neither a tool peak nor a fixed point peak was used to find the positions for the tool calculation. Using a measured point coordinate by the laser tracker, the reflector was mounted at the robot tool. After this, I moved the robot, so that the reflector was in the same 3D-coordinate again. I repeated this with all other needed points for the TCP calculation.

Using the laser trackers 3D-coordinates for reaching a desired position for the calculation of the tool transformation, is a more accurate way then the common peak-to-peak method., because the displayed deviations between current and desired reflector position are in the resolution range of 0.1 mm.

9.1.2 Gear factor calibration

The gear factor offset of the used linear track is influencing the exactness in measurement of the moved distance by the track skid on which the robot is mounted (see subchapter 4.2.3). A correction of the gear factor setting, integrated in the robot control avoids a deflection of the robots tool center point position depending on the current linear track position. During robot analyzations and improvements used for this thesis, I made an analyzation and correction of the gear factor error by measurement of the real moved distance of the linear track to the commanded distance set in the robot control. By this, I obtained that the linear increasing error during movements on the track skid caused by an incorrect gear factor was compensated.

9.1.3 Alignment identification

The correct integration of the movement direction of the linear track in the robots base frame is the base for an exact linear track profile identification, see subchapter 4.3. For improvement of this integration, I used highly accurate measurement data of the linear track direction. Based on this measurements, the alignment of the robot base frame to the track direction can be calculated more exactly than using the robot integrated calculation routine.

9.1.4 Base frame integration

An exact identification of the robots base frame is a significant step for improvement of the application accuracy and one of the in subchapter 3.3.1 mentioned primary influences. The base frame identification mentioned in 3.5.1, using multiple different robot TCP positions with following least-square-fitting algorithm, is used for the analysis of the linear track and is providing a powerful method for workobject identification.

9.2 Linear track modeling

The basis for the main contribution of this work, is the analyzation and modeling of the linear track. Figure 90 is presenting the parts of which my contribution for linear track modeling consists of.

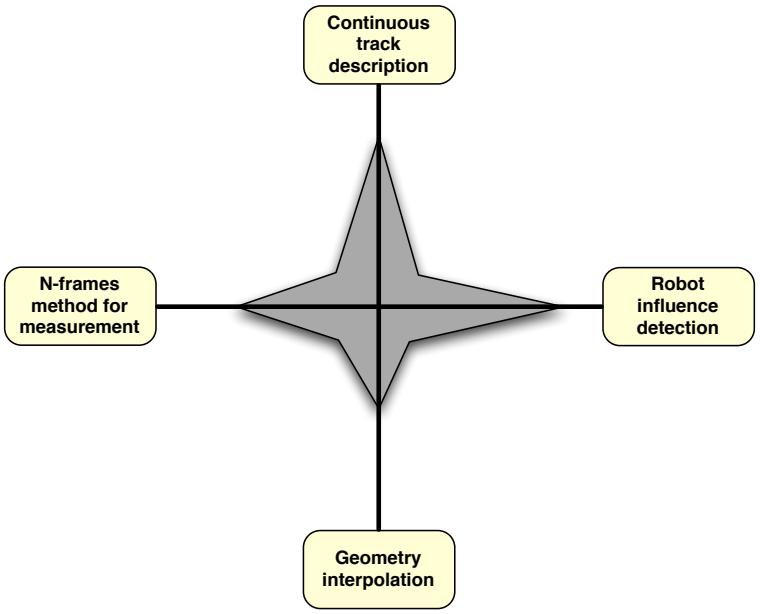

Figure 90: *Facets of linear track optimization*

N-frames method
The change from discrete 3D-measurements to a continuos 6D-description of the linear track I reached by a new method, using the identified base frames of the robot. In each fix sampling position of the skid on the track, I measured the robot in 10 different TCP positions (subchapter 4.3.2). This measurement data was used to calculate each robot base frame as input data for a cubic spline interpolation.

Geometry Interpolation
Needed for the continuous description, I interpolated the linear track geometry between the measured sampling positions (coordinate frames) in subchapter 4.6. An usable interpolation method was choosen and used for construction of the mathematical track description.

Continuous description
I obtained a continuous description by a new special technique for measurement (subchapter 4.3) and calculation of the track profile , depending on the current skid position. Due to the mathematical model of the track, I made it possible to describe the track deviation at an arbitrary skid position.

Robot influence detection
One big advantage of my new way of linear track modeling is the integration of the robots influence on the linear track deformations. Common techniques of linear track analyzation are not taking proper consideration to this influence. Due to the smaller gap between mathematical model and practical behavior, I obtained to enhance simulations of the robot application cells, needed for easier offline program construction.
While measuring the same TCP movement positions on different skid positions, the robot was used in his repeating accuracy, which is normally up to 10 times better than its absolute uncertainty. Disregarding the error influence of the repeating accuracy, for the measurements the robot is taken as a quasi rigid body.

9.3 Automated accuracy analyzation

For an easy identification of the robots static and dynamic accuracy, a program written in C++ was developed to work as an automated analyzation tool. The basis idea of this tool is to release the user from an exhausting evaluation of huge amounts of measurement data during quasi-continous scans of the robots TCP using sampling rates up to 1000 measurements per second. Figure 91 is showing the most important contributions, coming with this new software.

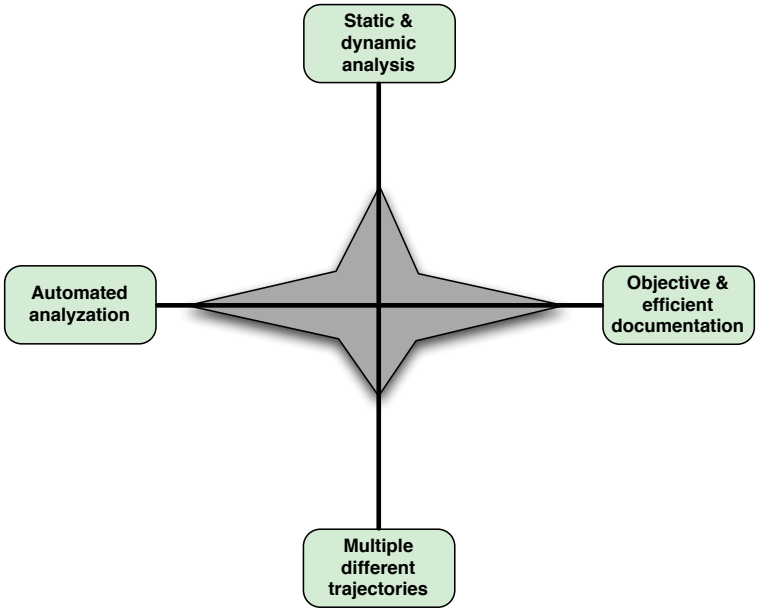

Figure 91: *Contribution facets of accuracy analyzation*

This tool is currently used by a big german car manufacturer for checking industrial robots during a pre-buy test. It is adapted to the special needs of car manufacturers to enable a professional automated and objective test routine for proofing the features promised by the robot manufacturers.

Multiple different trajectories
Besides of its main focus to straight movements, I realized additionally the analyzation of certain movement trajectories which are commonly used in automotive industry.

Static & dynamic analysis
An integrated analyzation of the static accuracy and even of the dynamic accuracy gives a new contribution to this software by realizing one complete software tool for all needed performance tests in evaluation phase.

Objective & efficient documentation
Due to the automated process of result and performance calculation during robot analyzation, I created an objective and easy-to-use system for the applicator. By this, the needed time for robot performance tests is decreased significantly.

Automated anayzation
The automated process of result documentation data is giving a possibility of batch creation of presentable material. There is no need for manual data adaption or preparation. Integrating numerical and graphical results of the analyzation, methods for improving the readability of the documentation are giving a new approach in taking rapid predictions about the quality of an industrial robot.

9.4 Offline track compensation

The main contributions of this thesis are represented by the offline robot program correction based on a compensation of the linear track (figure 92). This correction is an unique feature and is improving massively the application accuracy. Based on the track analyzations which were developed in this thesis, the continuos mathematical description of the linear track can be used to generate the needed correction vectors to compensate the linear track error influence.

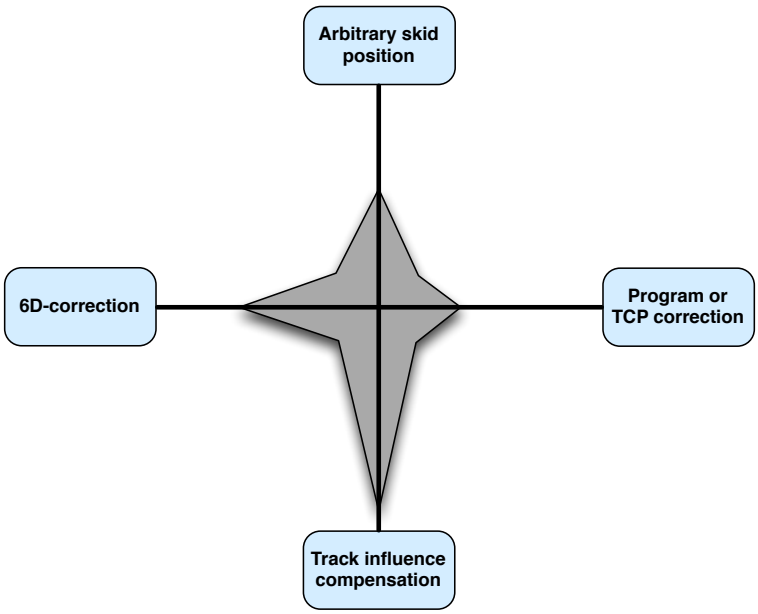

Figure 92: *Important Parts of the offline track compensation*

Track influence compensation
Based on the calculated model of the linear track, I was able to compensate the linear track error influence on the robots TCP (see subchapter 4.7.2). The resulting error minimization occurring after base frame identification, can be seen in table 8.

6D-correction
I obtained a track compensation in all six degrees of freedom by correction vectors represented in a 6-dimensional format (subchapter 7.3.4) to enable cartesian correction in x, y and z, but also an orientation correction in α, β and γ.

Arbitrary skid position
Including the robot influence on the track, a correction vector for an arbitrary robot skid position on the track can be calculated. For this, I realized an independence of the correction algorithm to the current position of the skid (see subchapter 7.3.3).

Program or TCP correction
The practical correction of the robot programs can be done in an offline way by modification of the program point coordinates or by an online correction during movement of the robot. While correcting the robot in online mode, the current robot TCP position is corrected using special features supplied by the robot control like correction offset register or base frame shifts for example.

In general, the correction of the robot TCP positions and therefore the optimization of the robot programs is a further step to reach a higher level of flexible automation (see results in subchapter 7.3.5). By reduction of the deviations between theoretical model and practical application of the robot, a higher level of offline robot programs is possible to generate. This higher level results in an easier adaption and on-site integration of the robot programs and a significant reduction in time and money.

9.5 Program mirror optimization

As an additional step for the creation of minimum effort during setup phase of modern robot applications, the transference of already corrected programs to other robots is one of the contributions of this thesis. Once corrected an offline created robot program regarding the current linear track profile, this program can normally not be copied or mirrored to other robots on different linear tracks.

Program re-calculation

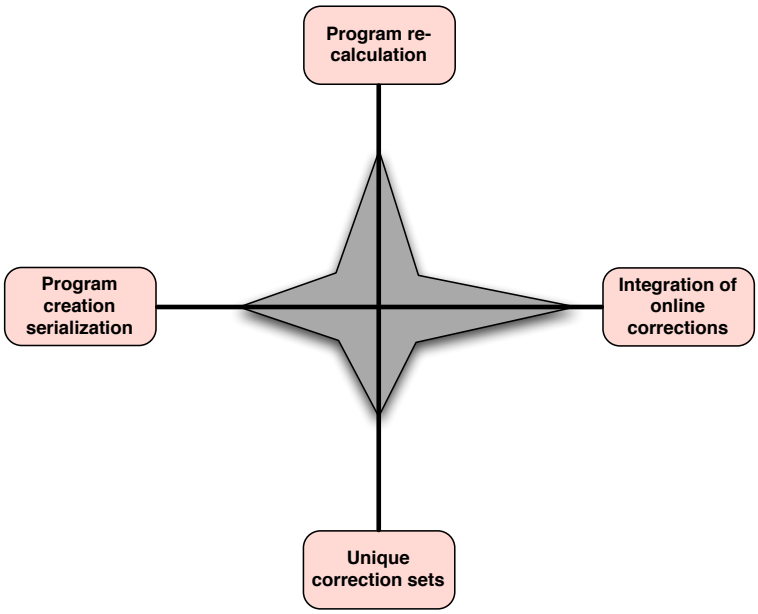

Figure 93: *Contributions in optimizing the copy and mirror process*

Using the new method which I realized in this thesis and which is presented in chapter 8, it is now possible to convert a robot program down to a "zero-corrected" program by inverse application of the track profile corrections.

Unique correction set
Adapted to each robot particularly, I realized an unique set of correction values integration the track compensation (subchapter 8.3). After transference of this programs to the next robot, the unique corrections values, can be applied.

Integration of online corrections

With this method, I obtained the possibility to copy and mirror even online-corrected robot programs including their linear track geometry. By this, it is now possible to copy a program after online correction to another robot with different linear track profile (see linear track part in subchapter 8.1).

Program creation serialization
The mirroring optimizations which I developed are the basis of an efficient serial creation of robot programs for usage in parallel robot applications to increase the production output. Time consuming online teaching processes and the resulting setup time for the robot application can now be reduced to lower lever.

10 Conclusion and further work

The improvement of the various parts of the robot application is causing in decreased uncertainty of the TCP position. This is minimizing the gap between theoretical model included in the CAD-system and real behavior of the robot.

Resulting the amount of robot positions, which have to be corrected manually during setup of the application is reduced significantly. This is a crucial and important step into the direction of automated robot program creation, which is a demand of flexible automation.

Further steps which are going along with the ideas and contributions of this thesis are the automated correction of the remaining offline created points, which are not usable for real application even after first correction. Using the laser tracker device for checking the current TCP position and its desired position on the object, is one possible realization of a fully automated system for offline-to-online adaption.

First prototype implementations of this system are already done and will be published after this thesis.

As a final conclusion, the correction methods for offline programs presented in this thesis, are completing the current state of the art in modern industrial robotics with a powerful and time-saving strategy for pushing the application efficiency.

Annex 1

Technical data of the LTD800

Device dimensions	Sensor: 220 x 280 x 875 mm Controller: 510 x 485 x 200 mm
Weight	Sensor: 34.1 Kg Controller: 17 Kg
Measurement principle	Two independent high-accurate angle encoders for horizontal and vertical angle encoding. High resolution laser interferometer for distance measurement.
Tracking speed	*Across beam direction:* Target speed: $\leq$ 4 m/s Acceleration: $\leq$ 2g *Along beam direction:* Target speed: $\leq$ 6 m/s Acceleration: ∞
Maximum range	$\leq$ 40 m in radius $\pm$235° horizontal $\pm$45° vertical
Measurement rate	$\leq$ 3000 points / sec measurement $\leq$ 1000 points / sec output
Environment conditions	Temperature: +0°C - +40°C Humidity: 10 - 90% (non-condensing) Altitude: 0 - 5000 m

Table 11: *Technical facts of Leica Laser Tracker LTD800*

Measurement accuracy

Interferometer

Distance resolution	1,26 μm
Absolute uncertainty	$\pm$0,5 μm/m additional birdbath uncertainty: 10 μm
Repeatability	$\pm$ 5 μm/m

Table 12: *Interferometer accuracy of LTD800*

Angle encoder

Angular resolution	0.14 arc sec
Absolute uncertainty	$\pm$ 25μm (non-moving target 0 - 2,5 m distance)
Repeatability	$\pm$ 12 μm (0 - 2,5 m distance)

Table 13: *Angular uncertainty of LTD800*

References

[1] Čapek, Karel; *R.U.R. - Rossum's Universal Robots*, Czech drama, 1921

[2] International Organization for Standardization, DIN EN ISO 9283:1998; *Manipulating industrial robots - Performance criteria and related test methods*, European Standard, May 1999.

[3] Kleinkes, M.; *A plan of a static and dynamic accuracy check for industrial robots*, Masterthesis at the University of Applied Science in Gelsenkirchen, Germany 2004.

[4] Wloka, D.W.; *Robotersysteme I - technische Grundlagen*, Springer, Germany 1992.

[5] Denavit, J., Hartenberg, R.S.; *Kinematic Synthesis of Linkages* , McGraw-Hill, USA, 1964.

[6] Denavit, J., Hartenberg, R.S.; *A kinematic notation for lower-pair mechanism based on matrices*, ASME journal of applied mechanics 22, 1955.

[7] Kreuzer, E.J., Meiflner H.-G., Lugtenburg, J.-B., Truckenbrodt, A., *Industrieroboter* Springer, Germany 1994.

[8] Faestermann, K.; *Genauigkeitsuntersuchung der Kalibrierung von offine-programmierten Roboterzellen* Masterthesis at the university of applied science in Gelsenkirchen, Germany 2004.

[9] Beyer, L., Wulfsberg, J.; *Practical robot calibration with ROSY*, Robotica Volume 22, Issue 5, p. 505-512, Germany, 2004.

[10] Craig, J.; *Introduction to robotics - Mechanics and Control*, Addison-Wesley, 1955.

[11] Spong, M.; *Robot Dynamics and Control*, John Wiley & Sons, 1989 .

[12] Press, W.; Teukolsky, S., Vetterling, W., Flannery, B., *Numerical recipies in C*, Cambridge University Press, 1992.

[13] Bronstein, I.N., Semendjajew, K.A., Musiol, G., Muehlig, H.; *Taschenbuch der Mathematik*, Verlag Harri Deutsch, 2000.

[14] Engeln-Muellges, G., Niederdrenk, K., Wodicka, R.; *Numerik-Algorithmen: Verfahren. Beispiele, Anwendungen*, Springer Berlin, 2004.

[15] Kleinkes, M.; *Fehlerberechnung zwischen linearer und kubischer Spline Interpolation von diskret gemessenen Roboterflansch-Koordinaten*, Report at the University of Applied Science in Gelsenkirchen, 2006.

[16] v. Grueningen, D.; *Digitale Signalverarbeitung*, Fachbuchverlag Leipzig, 2002.

[17] Balzert, H.; *Lehrbuch der Software-Technik*, Spektrum Verlag, 1996.

[18] Wilms, G; *C++ - Das Grundlagenbuch*, Data Becker, 1997.

[19] National Institute of Standards and Technology, NIST; *Technical Note 1297: Guidelines for Evaluating and Expressing the Uncertainty of NIST Measurement Results*, 1993.

[20] International Organization for Standardization, ISO; *Guide to the expression of uncertainty in measurements*, 1994.

[21] Montgomery, D.C., Runger, G.C.; *Applied statistics and probability for engineers*, John Wiley & sons, 2003

[22] Leica Geosystem AG, *www.leicageosystems.com*.

[23] Markendorf, A.; *Uncertainty analysis of spatial distance measurements*, Leica Geosystems AG, Switzerland, 2004.

[24] Koenig, A.; *Analyse der Spiegelbarkeit asymmetrischer Roboter-Werkzeuge und Software-Implementierung zur praktischen Anwendung*, Diploma thesis at the university of applied science in Gelsenkirchen, Germany 2008.

[25] Li, X.; *Entwicklung eines Programms zur Online-Positionskorrektur eines Industrieroboters unter Verwendung verschiedener Interpolationsmethoden*, Masterthesis at the university of applied science in Gelsenkirchen, Germany 2007.

[26] Klekot, M.; *Fehlerfortpflanzungsanalyse einer offline-programmierten Roboterzelle*, Diploma thesis at the university of applied science in Gelsenkirchen, Germany 2007.

[27] Kleinkes, M., Lilienthal, A., Neddermeyer, W.; *Highly accurate integration of track motions* International conference on Informatic in Control, Barcelona, 2005.

[28] Kleinkes, M., Neddermeyer, W., Schnell, M.; *An automated quick accuracy and output signal check for industrial robots*, WSEAS Int. Conf. on ROBOTICS, CONTROL and MANUFACTURING TECHNOLOGY (ROCOM '06), Hanghzou (China), 2006.

[29] Kleinkes, M., Ignea, A., Neddermeyer, W., Schnell, M.; *Interpolation of linear track movements of modern industrial robots*, transactions of electronics and telecommunication, Polytechnical University Timisoara, Romania, 2006.

[30] Kleinkes, M., Lilienthal, A., Winkler, W.; *Static and dynamic accuracy and process capability tests for industrial robots*, WSEAS International Conference, Orlando (USA), 2005.

[31] Kleinkes, M., Neddermeyer, W., Schnell, M.; *Improved method for highly accurate integration of track motions*, International conference on Informatic in Control, Portugal, 2006.

[32] Kleinkes, M., Neddermeyer, W., Schnell, M.; *Linear track accuracy improvement using cubic spline interpolation*, 5th international symposium on robotics and automation, Hidalgo (Mexico), 2006.

[33] European co-operation for accreditation; *Expression of the uncertainty of measurement in calibration*, publication reference EA-4/02, 1999.

[34] Philips, S. D.; *Traceability, calibration and measurement uncertainty issues regarding coordinate measuring machines and other complex instruments*, International dimensional workshop, Knoxville (USA) 2000.

[35] Ortmaier, T., Hirzinger, G.; *Cartesian control of robots with working-position dependent dynamics*, Deutsches Institute fuer Luft- und Raumfahrt - Institute of robotics and mechatronics.

[36] Convay, J., Smith, D.; *On quaternions and octonions*, A K Peters, Massachusetts, 2003.

[37] Dautray, R.; *Mathematical analysis and numerical methods for science and technology*, Springer, London 2000.

[38] Maas, H. G.; *Dynamic photogrammetric calibration of industrial robots*, SPIE proceedings series Vol. 3174, 1997.

[39] Nitschke, H.; *Zur Bestimmung geometrischer Parameter von Industrierobotern*, PhD thesis at the Bavarian Academy of Sciences, Munich (Germany), 2002.

[40] Sallal Yassin A.; *A method for path- and orientation-leading for industrial robots under the aspect of general spacious pathes* PhD thesis at the Technical University Chemnitz-Zwickau, Germany, 1995.

[41] Daemi-Avval, M.; *Modeling and identification of the dynamic of industrial robots for the use in closed loop controls*, Progress-Reports VDI, Series 8, Volume 719, Germany, 1998.

[42] Angeles, J.; *Fundamentals of robotic mechanical systems*, Springer, Germany 1997.

[43] Bachman, G., Narici, L., Beckenstein, E.; *Fourier and wavelet analysis* Springer New York, 2002.

[44] Bracewell, R.; *The fourier transform and its applications.*, McGraw-Hill, Boston (USA), 2000.

[45] Grupp, F.; *MATLAB 7 fuer Ingenieure* Oldenburg Verlag, Germany, 2004.

[46] Gramlich, G., Werner, W.; *Numerische Mathematik mit MATLAB*, DPunkt Verlag, Germany, 2000.

[47] Quateroni, A., Saleri, F.; *Wissenschaftliches Rechnen mit MATLAB*, Springer Verlag, 2006.

[48] Scheibl, H.-J.; *Numerische Methoden f§r den Ingenieur*, 2nd edition, Expert Verlag, Germany, 1994.

[49] Huckle and Schneider; *Numerik fuer Informatiker*, Springer Verlag, Germany, 2002.

[50] Plato, R.; *Numerische Mathematik kompakt*, 2nd edition, Vieweg Verlag, Germany, 2004.

[51] Rice, J. R.; *Numerical methods, software and analysis*, 2nd edition, Academic Press Inc., 1993.

[52] Carnahan, B.; *Applied numerical methods*, Robert Krieger Publishing Company, 1990.

[53] Antia, H. M.; *Numerical methods for scientists and engineers*, 2nd edition, Birkhauser Verlag, Germany, 2002.

[54] Davis, P. J.; *Interpolation and approximation*, Dover Publications Inc., 1975.

[55] Conte, S. D., Boor, de C.; *Elementary numerical analysis - an algorithmic approach*, Third edition, McGraw-Hill, 1980.

[56] Werner, M.; *Digitale Signalverarbeitung mit MATLAB*, Vieweg Verlag, Germany, 2003.

[57] Achilles, D.; *Die Fourier-Transformation in der Signalverarbeitung*, Springer Germany, 1985.

[58] Ramanathan J.; *Methods of applied Fourier Analysis*, BirkhŁuser, Germany, 1998.

[59] Meyer, M.; *Signalverarbeitung*, Vieweg, Germany, 1998.

[60] Grueningegn, v. D.; *Digitale Signalverarbeitung*, Fachbuchverlag Leipzig, 2002.

[61] Mertins, A.; *Signaltheorie*, Teubner Verlag, Germany, 1996.

[62] Brigham, E.O.; *FFT-Anwendungen*, Oldenburg Verlag, Germany, 1997.

[63] Rupprecht, W.; *Signale und Ębertragungssysteme*, Springer Germany, 1993.

[64] Kreuzer, E.J., Meissner, H.-G., Lugtenburg, J.-B., Truckenbrodt, A.; *Industrieroboter*, Springer Germany, 1994.

[65] Hesse, S.; *Industrieroboterpraxis*, Vieweg-Verlag, Germany, 1998.

[66] Lehmann, R.; *AC-Servo-Antriebstechnik*, Franzis, Germany, 1990.

[67] Craig, J.; *Introduction to Robotics - Mechanics und Control 2. Edition*, Addison-Wesley, 1989.

[68] Kocheali, H., Nowrouzi, A., Kavina, Y.B., Whitaker, R.A.; *Factors affecting robot performance*, Industrial robot, Vol. 18 No. 1, pp. 9-13, 1991.

[69] Spong, M.W., Vidyasagar, M.; *Robot dynamics and control*, John Wiley & Sons, 1989.

[70] Wloka, D.W.; *Robotersysteme 1 - Technische Grundlagen*, Springer Berlin, 1992.

[71] Fu, K.S., Gonzalez, C.S.G.; *Robotics - mechanics and control*, Addison-Wesley, 1955.

[72] Paul, R.P., Shimano, B.E., Mayer, G.; *Kniematic control equations for simple manipulators*, IEEE transactions on systems and cybernetics, Vol. SMC-11 no. 6, 1981.

[73] Unbehauen, H.; *Regelungstechnik I*, 12th edition, Vieweg Verlag, Germany, 2002.