2013 IEEE/ASME International Conference on
Advanced Intelligent Mechatronics (AIM)
Wollongong, Australia, July 9-12, 2013

3D Mapping using a ToF Camera for Self Programming an Industrial Robot*

N. Larkin, Z. Pan, S. Van Duin, J. Norrish

Abstract— **Automated Offline Programing (AOLP) is a cost effective robot programming method. However, it relies on accurate CAD information of the work environment to perform optimally. Incorrect CAD data is a known source of error for AOLP systems. This paper introduces a new sensor based method of programming that extends the concept of AOLP. Using a ToF camera to map the environment, there is no reliance on CAD data. The problem of motion planning to efficiently map the environment is examined and changes to the motion planning algorithm are proposed and tested.**

I. INTRODUCTION

The lead-through programming method is still one of the most widely used programming techniques for industrial robots. Although simple, lead-through programming methods can be time-consuming and expensive for certain applications such as robotic arc welding [1].

An alternative to lead-through programming methods is Offline Programming (OLP). OLP is conducted in a computer-simulated environment of the robotic work cell using CAD models of the environment and work objects. The advantage of OLP over online programming methods such as lead-through programming is that it does not require the actual robot. However OLP does little to reduce programming overhead, instead moving effort from the shop floor to the computer environment.

Automated Offline Programming (AOLP) reduces programming effort further by automating many of the manual functions required by conventional OLP. Process start and end points (e.g. arc-on and arc-off locations) are automatically generated through analysis of CAD geometry. Motion planning algorithms generate collision free paths. Translators convert motions and process settings into code that is executed by the robot controller. In the case of robotic arc welding, several research projects [2][3] and commercial systems exist. Kranendonk Production Systems indicate that 1 hour of programming can generate 15 hours of welding in their AOLP product RinasWeld [4].

The problem for AOLP systems is the reliance on accurate CAD geometry. AOLP relies on the assumption that the work environment is modeled accurately in its entirety. AOLP conducted by the authors for industry has demonstrated that incorrect CAD information is a key source of error for AOLP systems. These errors often go unnoticed until a robot collision occurs.

This paper introduces a method of programming that extends the concept of AOLP to remove the reliance of CAD information. An additional sensor added to the robot is used to map the work environment to generate the information usually obtained from CAD data. This enables a form of self programming, where the robot is able to perform a task without conventional programming. The focus of this paper is 3D mapping using a ToF camera, in particular the motion planning algorithm where modifications to a generic algorithm are proposed and examined.

II. SENSOR BASED AOLP

In order to map the work environment, a sensor is added to the robot system enabling the environment geometry to be acquired. The sensor used is a commercially available Time-of-Flight (ToF) camera.

Utilizing ToF camera technology for 3D mapping is not new. Work in this area has been ongoing for nearly decade, for example [5][6][7], with the majority focused on mobile robot platforms. This paper focuses on industrial robots, characterized by 6-DoF serial link manipulators with a fixed base. In particular, adapting the motion planning algorithm for 3D mapping.

The ToF camera has several advantages over other vision systems. The camera used, a MESA Imaging SwissRanger™ SR4000, is compact, has a high frame rate, and has a wide field-of-view (FoV). The main disadvantage is the poor accuracy. Robotic Gas Metal Arc Welding (GMAW) requires accuracy to within 1.5mm [8]. A ToF camera alone is not capable of achieving this accuracy, so secondary calibration techniques must be employed. Specifications of the SwissRanger™ SR4000 are listed in Table 1.

*Research supported by the Defence Material Technology Centre (DMTC) under projects 2.4 and 3.5.

N. Larkin is with the University of Wollongong, Wollongong, NSW 2522 Australia (+61 2 4221 5283; email: nlarkin@uow.edu.au)

Z. Pan is with the University of Wollongong, Wollongong, NSW 2522 Australia email: (zengxi@uow.edu.au)

S. Van Duin is with the University of Wollongong, Wollongong, NSW 2522 Australia (email: svanduin@uow.edu.au)

J. Norrish is with the University of Wollongong, Wollongong, NSW 2522 Australia (email: johnn@uow.edu.au)

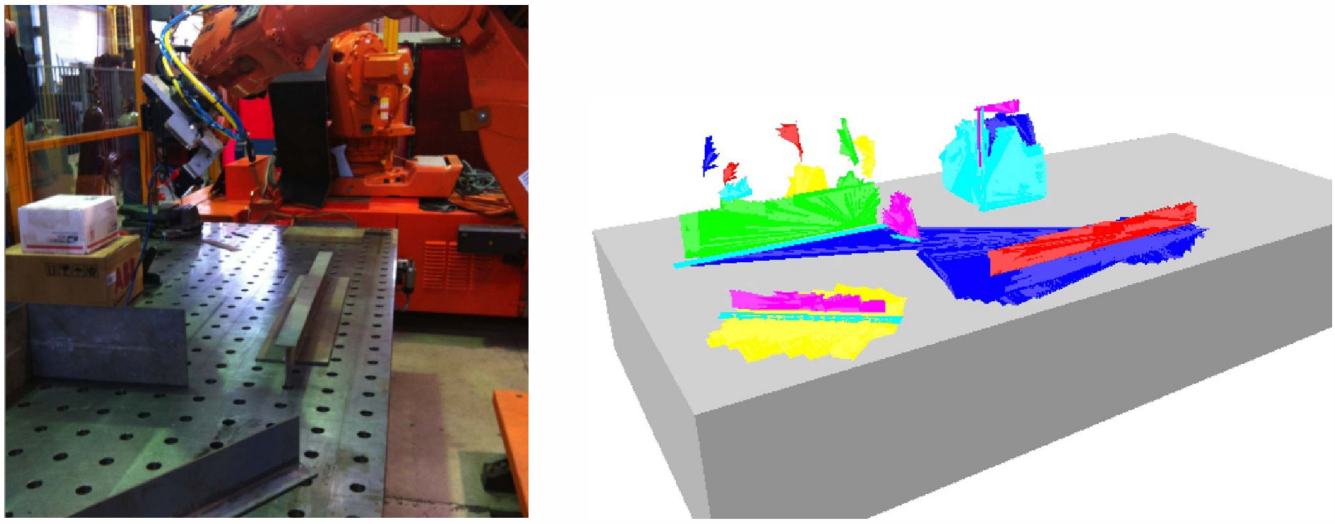

Figure 1. Geometry of plates to be welded. (a) Picture of plates to be welded and (b) plates as detected by the Sensor Based sensor system.

This concept can be applied to a wire range of robot applications. In Fig. 1, the sensor based AOLP system is being used to fillet weld plates together.

III. Mapping the Work Environment

Simple filtering techniques and a method to join each data set gathered from the ToF camera is used to map the environment. Many other researchers have developed sophisticated 3D mapping techniques for ToF cameras [9][10] and show that the raw output from the camera can be greatly optimised with calibration [5][11][12]. However, absolute mapping accuracy is not the focus of this work.

TABLE I. ToF Camera Specification

Specifications of the SwissRanger™ SR4000 [13]			
Parameter	*Notation*	*Value*	*Unit*
Horizontal FoV	FoV_h	69	°
Vertical FoV	FoV_v	56	°
Number of Horizontal Pixels	p_h	176	
Number of Vertical Pixels	P_v	144	
Size		65x65x76	mm

For mapping purposes, the work environment is represented by the three-dimensional matrix E as follows:

$$E = \left\{ e_{ijk} \in \mathbb{Z}^3 \mid i = 1, \dots, \frac{s_x}{\Delta}\ j = 1, \dots, \frac{s_y}{\Delta}\ k = 1, \dots, \frac{s_z}{\Delta} \right\}. \quad (1)$$

Δ is the sampling distance and $s_x, s_y, and\ s_z$ are the size of the work environment in each direction. The value of e_{ijk} indicates the state of the work environment at the location $\Delta[i \quad j \quad k]^T$. Negative values are used to indicate object free space, and positive values indicate that an object exists in the location. E is initialized to zero representing an unknown state.

$$E = \{0\} \quad (2)$$

The output of the ToF camera is a two-dimensional matrix I, representing the distance to the detected object for each pixel. Amplitude and confidence data is also gathered from the data and used to remove erroneous data from I.

$$I = \{ i_{hv} \in \mathbb{R}^2 \mid h = 1, \dots, p_h\ v = 1, \dots, p_v \} \quad (3)$$

where p_h is the number of pixels in the horizontal direction and p_v is the number of pixels in the vertical direction.

I forms the object D', by converting the distance i_{hv} into a two-dimensional group of vectors representing the state of the work environment.

$$D' = \left\{ d_{hvs} \in \mathbb{Z}^3 \mid d'_{hvs} = \begin{cases} -1, & s < i_{hv} \\ 2, & s = i_{hv} \\ 0, & s > i_{hv} \end{cases}, s = 0, \Delta, 2\Delta, \dots, \infty \right\} \quad (4)$$

Using the rotation matrix relevant to each pixel the transformation matrix relevant for d'_{hvs} can be calculated.

$$rot_{hv} = \begin{bmatrix} \cos(\theta_v) & \sin(\theta_h)\sin(\theta_v) & \cos(\theta_h)\sin(\theta_v) \\ 0 & \cos(\theta_h) & -\sin(\theta_h) \\ -\sin(\theta_v) & \sin(\theta_h)\cos(\theta_v) & \cos(\theta_h)\cos(\theta_v) \end{bmatrix} \quad (5)$$

where

$$\theta_h = \left(h - \frac{1}{2}\right)\frac{FOV_h}{p_h}\ ,\ \theta_v = \left(v - \frac{1}{2}\right)\frac{FOV_v}{p_v}, \text{ and} \quad (6)$$

$$^{pixel}_{camera}T_{hv} = \begin{bmatrix} & rot_{hv} & & \begin{matrix}0\\0\\0\end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix},\ ^{object}_{pixel}T_s = \begin{bmatrix} I & & & \begin{matrix}0\\0\\s\end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (7)$$

With the addition of the robot pose transformation matrix $^{tool}_{world}T$ and the camera mounting transformation $^{camera}_{tool}T$. the 3D object D that represents the work environment data detected by the camera is calculated using

$$D = \left\{ c_{ijk} \in \mathbb{Z}^3 \mid c_{ijk} = d'_{hvs} \right\} \quad (8)$$

where

$$\Delta \begin{bmatrix} i \\ j \\ k \\ 1/\Delta \end{bmatrix} = {}^{tool}_{world}T \cdot {}^{camera}_{tool}T \cdot {}^{pixel}_{camera}T_{hv} \cdot {}^{object}_{pixel}T_s \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \quad (9)$$

S is incorporated into the work environment using

$$E = E + D. \quad (10)$$

IV. Mapping an Environment with a Serial Link Robot

In the AOLP process, the motion planner generates a collision free path from one location to another. Motion planning is usually conducted in C, the configuration space of the robot. For six-degree-of-freedom (DoF) robots, C is a six dimensional matrix where each dimension corresponds to a joint angle of the robot. C is split into two sections; C_{free} and $C_{forbidden}$ that refer to the free space and space occupied by obstacles respectively. In this application, a third state $C_{unknown}$ is introduced. $C_{unknown}$ refers to the configuration space that is yet to be mapped. To avoid potential collision, robot paths must not intersect $C_{forbidden}$ or $C_{unknown}$.

Initially, the only known free configuration space is the space in which the robot occupies. The known work environment is also limited to that of the initial robot geometry and the first image from the camera. Therefore, unless the camera has a complete view of the robot, there will not be any motion paths that do not intersect $C_{forbidden}$ or $C_{unknown}$. To work around this, a small area around the robot is assigned as known unobstructed space.

The first task for the motion planner is to map the work environment. To achieve this, two changes to the motion planning algorithm are proposed:

1. Record the intersection position when intersection between a path and $C_{unknown}$ occurs.
2. Incorporate a model of the ToF camera to predict the volume of unknown space that can be detected.

The motion planner used is based around the common Probabilistic Roadmap Planner (PRM) that utilizes random sampling of the free space to interconnect robot positions [13]. The encompassing idea of a PRM is to randomly sample C_{free}. These samples, known as nodes, are tested for connection and the process is looped until a path from start to goal is found.

When mapping an unknown environment however, the goal of the motion planner is to maximize the detection of unknown space by the ToF camera. The secondary goal is to map unknown space around the robot to maximize the motion paths available. The local planner uses joint space interpolation between robot configurations. The algorithm starts at a location located within the PRM tree QT_n and interpolates a path towards Q_{test}, the point being tested for connection. If an intersection occurs between the path and $C_{forbidden}$ or $C_{unknown}$ the connection and Q_{test} are discarded. In the case that the intersection occurred with $C_{unknown}$, the intersection position is recorded in U using

$$U = \{u_{ijk} \in \mathbb{Z}^3 \mid u_{ijk} = u_{ijk} + 1\}. \quad (11)$$

Pseudo code for the local planner is listed in Aligorithm 1.

If the local planner successfully connects Q_{test} to QT_n, Q_{test} is added to QT and the amount of unknown space that may be detected by the camera, M_{score}, is calculated. The value of unknown space found to limit robot movement is increased to encourage mapping around these areas.

$$M_{score} = \sum_{h=0}^{p_h} \sum_{v=0}^{p_v} \sum_{s=0}^{s_{max}} w\left(1 + o_{coeff} u_{ijk}\right) \quad (12)$$

where

$$w = \begin{cases} 0, & e_{ijk} \le k_f \\ 1, & k_f < e_{ijk} < k_o \\ 0, & e_{ijk} \ge k_o \end{cases}, \quad (13)$$

o_{coeff} is the obstacle weighting coefficient for U, and $[i \quad j \quad k]^T$ is calculated using eq. (9). k_f and k_o are the threshold values for free and obstructed space. The summing function $\sum_{s=0}^{s_{max}}$ is halted when $e_{ijk} \ge k_o$ occurs as the camera cannot detect space behind known obstacles. Algorithm 2 lists the pseudo code for the modified motion planner.

Algorithm 1 Modified Local Planner
Input Q_{test}, QT, n, U
Output U, $B_{connect}$

```
Q_path ← interpolate from QT_n to Q_test
for all Q_path
    if Q_path ∩ (C_forbidden ∨ C_unknown)
        B_connect ← false
        if Q_path ∩ C_unknown
            U ← u_ijk + 1
        exit for
    if Q_path = Q_test
        B_connect ← true
```

Algorithm 2 Modified Motion Planner
Input QT
Output QT, QP_{next}

```
Q_test ← i random configurations in C_free
for all Q_test
    if local planner finds connection between Q_test ← Q_n
        Q_n ← score Q_n
        QT ← Q_n
    else
        discard Q_n
Q_next ← search QT for max M_score
QP_next ← list of connected nodes from Q_current to Q_next
```

V. Results

To test the performance of the proposed approach a simulation environment was developed in MATLAB. Although not suited for testing recursive algorithms such as PRMs, MATLAB was chosen for the simplicity of development and compatibility with existing robotic functions. Experience has shown that a 100-fold increase in speed can be obtained by porting repetitive MATLAB code to computer languages such as c++. It is for this reason that algorithm efficiency in terms of execution time was not investigated.

The simulation was based on an existing industrial robot work cell. The work cell comprises of an ABB IRB120 with a mounted SwissRanger™ SR-4000 ToF camera (see Fig. 2). Details of the work cell related to the simulation are listed in Table 2. An empty camera frame was taken and used as the camera data for all shots in the simulation. Robot movement was generated using the motion planner and solved using the kinematic model of the IRB120. Table 2 lists the common

simulation parameters. The robot base was placed at location [1000 1000 0].

The simulation was split into camera frame iterations as shown in Algorithm 3. For each test, 100 iterations were executed. The amount of mapped space was recorded at each step as a percentage.

TABLE II. SIMULATION PARAMETERS

Parameters used for Simulation			
Parameter	*Notation*	*Value*	*Unit*
Sampling Distance	Δ	20	mm
Work Environment Size in X-direction	s_x	2000	mm
Work Environment Size in Y-direction	s_y	2000	mm
Work Environment Size in Z-direction	S_z	1000	mm
Camera Sensing Distance		600	mm
Threshold for Free Space	k_f	-10	
Threshold for Obstructed Space	k_o	10	

Figure 2. Robot and work environment simulated. The robot is shown in the starting configuration.

Algorithm 3
Input: $E \leftarrow \{0\}$
Output: E, QT

1: $E \leftarrow \{0\}$, $QT \leftarrow \emptyset$, *iteration* $\leftarrow 0$
2: **loop** iteration++
3: $I \leftarrow$ data set from ToF camera
4: $E \leftarrow E + C$
5: **for all** QT
6: score Q_n
7: $QP_{next} \leftarrow$ **execute** modified motion planner
8: move robot to Q_{next}

A. Performance Comparison

Fig. 3 shows the difference in mapped space between the standard motion planner and the modified motion planner. Each planner was run 5 times with the average performance of each shown on the graph. A value of 100 was used for o_{coeff}. As expected, the modified motion planner was able to map the space more efficiently in terms of known space detected per iteration.

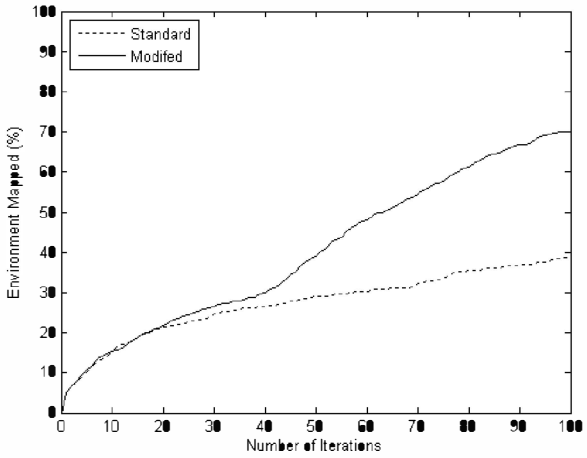

Figure 3. Average mapping perforamce of standard and modified motion planner.

Both planners show similar mapping performance from iteration 0 to iteration 40. The performance of the planners diverges from iteration 40 where the standard planner performs poorly. At iteration 100, the modified planner on average maps 70% of the environment; almost double that of the standard planner at 39%.

The divergence appears to be caused by two factors:

1. The freedom of the motion planner to plan paths around the environment.
2. The target Q_{next} chosen by the motion planner to take the images from.

B. Freeing the Robot of Unknown Space

A critical task for the motion planner is to identify the area around the robot as either free or forbidden. Left unmapped, this area will limit the paths available to the motion planner. A method to quantify the extent of C_{free} is to analyze the range of motion available to the motion planner. In this case, the extent of movement for joint 1 of the robot is expressed as a percentage of the full range available. The motion data is gathered from QT, the PRM nodal tree.

The range of motion against the number of iterations for a single test of each planning method is shown in Fig. 4. It indicates the performance advantage of the modified algorithm to free the robot.

The obstacle coefficient o_{coeff} is used to change the behavior of the modified planner by increasing the value of unknown space limiting robot movement. Fig. 5 shows the effect of this coefficient has on freeing the robot. Iterations required to obtain certain levels of Joint 1 freedom are listed in Table 3.

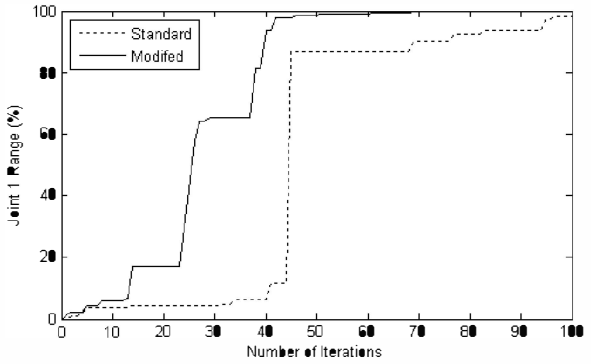

Figure 4. Plot of Joint 1 motion range from samples in the PRM tree QT against number of iterations.

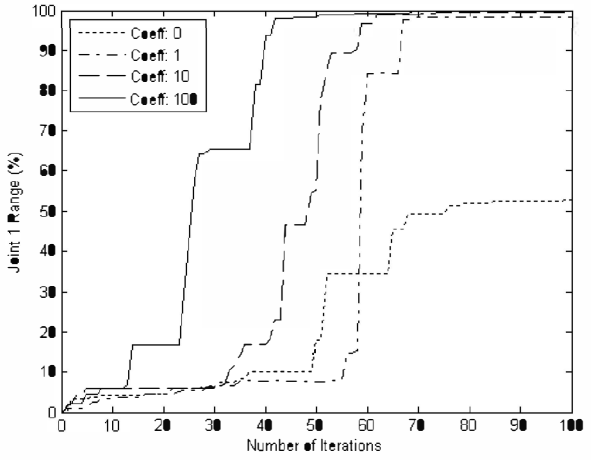

Figure 5. Plot of Joint 1 motion range from samples in the PRM tree QT for varying obstacle coefficients o_{coeff}.

TABLE III. COMPARISON OF JOINT 1 FREEDOM FOR VARYING OBSTACLE COEFFICIENTS

Coefficient O_{coeff}	Iterations Required to achieve Joint 1 Freedom			
	10%	*20%*	*50%*	*90%*
0 (inactive)	38	53	77	-
1	57	60	60	68
10	34	43	50	59
100	15	25	27	41

The data shows a correlation between the obstacle weighting coefficient and the number of iterations required achieving joint 1 freedom. To achieve half joint freedom, 50 fewer iterations are required when using a coefficient of 100 compared to when the obstacle weighting is deactivated. This improvement results is higher mapping performance as shown in Fig. 6.

Although mapping performance of the motion planning algorithm is improved, there is far more scope for optimization. Fig.6 shows that performance degrades as the known work environment approached 70%. It is believed that this is due to the poor distribution of random robot configurations generated by the PRM. This is a common problem when using uniform random distributions as shown in [14] and is a future area of investigation.

These results show that the motion planner is capable of generating locations for an industrial robot to map the work environment around it. This information can be used as an input to the AOLP process to generate robot programs automatically, without CAD information. In the case of arc welding, this system applied to an ABB IRB4400 has successfully been used to perform welds on simple geometry.

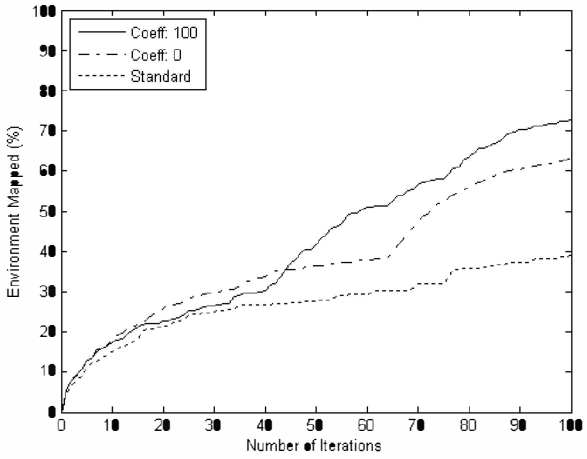

Figure 6. Mapping performance of standard, and modified motion planning algorithms with obstacle coefficient activated and deactivated.

V. CONCLUSION

This paper introduced the concept of AOLP using a sensor to map the work environment. The environment geometry is generated using data gathered from a ToF camera mounted to the robot being programmed. Modifications to the motion planning algorithm were made to improve mapping performance and the following conclusions can be made:

1. A random approach to mapping using a standard PRM does not result in high mapping performance.
2. The introduction of a calculation to score the unknown area detectable by the camera improved performance.
3. Using a high value for the obstacle coefficient reduced the number of iterations required to map the area around the robot.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of the Defence Materials Technology Centre (DMTC). The DMCT was established and is supported under the Australian Government's Defence Future Capability Technology Centres Program.

REFERENCES

[1] Pan, Zengxi, Joseph Polden, Nathan Larkin, Stephen Van Duin, and John Norrish. "Recent progress on programming methods for industrial robots." *Robotics and Computer-Integrated Manufacturing* 28, no. 2 (2012): 87-94.

[2] Ames, Arlo L., Elaine M. Hinman-Sweeney, and John M. Sizemore. "Automated generation of weld path trajectories." In *Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing, 2005.(ISATP 2005). The 6th IEEE International Symposium on*, pp. 182-187. IEEE, 2005.

[3] Polden, Joseph, Zengxi Pan, Nathan Larkin, Stephen Duin, and John Norrish. "Offline programming for a complex welding system using DELMIA automation." *Robotic Welding, Intelligence and Automation* (2011): 341-349.

[4] Anonymous. "Panel & Web Welding Gantry" Retrieved 15/12/2009, 2009, from www.kranendonk.com.

[5] May, Stefan, David Droeschel, Dirk Holz, Stefan Fuchs, Ezio Malis, Andreas Nüchter, and Joachim Hertzberg. "Three‐dimensional mapping with time‐of‐flight cameras." *Journal of field robotics* 26, no. 11‐12 (2009): 934-965.

[6] Weingarten, Jan W., Gabriel Gruener, and Roland Siegwart. "A state-of-the-art 3D sensor for robot navigation." In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3, pp. 2155-2160. IEEE, 2004.

[7] May, Stefan, David Droeschel, Dirk Holz, Christoph Wiesen, and Stefan Fuchs. "3D pose estimation and mapping with time-of-flight cameras." In *International Conference on Intelligent Robots and Systems (IROS), 3D Mapping workshop, Nice, France*. 2008.

[8] Weston, John. "Exploiting robots in arc welded fabrication." *The Welding Institute, Abington Hall, Abington, Cambridge CB 2 4 BQ, UK, 1989. 182* (1989).

[9] Prusak, A., O. Melnychuk, H. Roth, and I. Schiller. "Pose estimation and map building with a Time-Of-Flight-camera for robot navigation." *International Journal of Intelligent Systems Technologies and Applications* 5, no. 3 (2008): 355-364.

[10] May, Stefan, David Dröschel, S. Fuchs, Dirk Holz, and A. Nuchter. "Robust 3D-mapping with time-of-flight cameras." In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 1673-1678. IEEE, 2009.

[11] Kahlmann, Timo, Fabio Remondino, and H. Ingensand. "Calibration for increased accuracy of the range imaging camera swissrangertm." *Proc. of IEVM* (2006).

[12] Lindner, Marvin, and Andreas Kolb. "Calibration of the intensity-related distance error of the PMD ToF-camera." In *Proc. SPIE, Intelligent Robots and Computer Vision*, vol. 6764, p. 67640W. 2007.

[13] Anonymous. "SR4000 Data Sheet" Retrieved 09/02/2013, from http://www.mesa-imaging.ch/prodview4k.php.

[14] Kavraki, Lydia E., Petr Svestka, J-C. Latombe, and Mark H. Overmars. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces." *Robotics and Automation, IEEE Transactions on* 12, no. 4 (1996): 566-580.

[15] Boor, Valérie, Mark H. Overmars, and A. Frank van der Stappen. "The gaussian sampling strategy for probabilistic roadmap planners." In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 2, pp. 1018-1023. IEEE, 1999.