Closing the Gap between Industrial Robots and Underwater Manipulators

Satja Sivčev, Joseph Coleman, David Adley, Gerard Dooly, Edin Omerdić, Daniel Toal
MaREI – Marine Renewable Energy Ireland
University of Limerick
Limerick, Ireland
satja.sivcev@ul.ie

***Abstract*—Underwater robot manipulators mounted on work class ROVs are extensively used in a wide range of subsea intervention applications. Unlike the automated stationary robot arms used in factories, commercial underwater manipulators are tele-operated by human pilot in the loop. In this paper we describe investigations, development and adaptation of robot manipulator servo control approaches common for light assembly tasks in industry and the transfer of these techniques to marine robotics.**

***Keywords*—underwater manipulator, underwater intervention, closed-loop inverse kinematics**

I. Introduction

ROV technology has been the workhorse of sub-sea operations in offshore oil and gas industry for decades. It also plays an important role in marine renewable energy (MRE) and marine civil engineering industries as well as in marine science, military applications, etc. Intervention operations specific for inspection, repair and maintenance (IRM) services are performed by manipulators mounted on work class ROVs. Compared to stationary robot manipulators common in industrial robotics applications, commercial underwater manipulators are noticeably less advanced in the sense of autonomy. Therefore, our goal is to enhance the subsea manipulation capabilities and bring them closer to a comparable level with industrial manipulation. Hence, in this paper we describe investigations, development and adaptation of robot manipulator servo control approaches common for light assembly tasks in industry and the transfer of these techniques to marine robotics.

In Section II we describe the capabilities of industrial robotics, compare them with state of the art of subsea manipulation technology and briefly present work done by other authors as well as our goal for this project. Section III presents the analysis of the underwater manipulation scenarios to be addressed and gives a brief description of recent developments in this space. Section IV covers the algorithms used for software developed up to date and in Section V we conclude and describe future steps which are to be taken.

II. Background

Industrial robotics makes extensive use of servo controlled robot manipulators and control / programming environments with full kinematic engines (implementing forward and inverse kinematics), enabling automatic motion control to follow detailed robot motion control programmes with manipulators interacting with target(s). With integration of smart sensor systems, such as vision systems and visual servoing techniques, industrial robotics can deal with significant variability in the target object presentation, position, orientation, colour, etc., while addressing these target objects in automatic program operation [1]. Light assembly servo controlled robots are predominantly electrically powered.

Robot manipulators are extensively used in work class submarine ROVs for a variety of sub-sea tasks in different applications within offshore oil and gas, MRE and marine civil engineering industries as well as in marine science and military applications. As they are being used in a wide range of applications, underwater manipulators are designed for different purposes. There are manipulators equipped with grippers with limited mobility for lifting large, heavy objects, manipulators for fixing a detachable gripper to a selected, sunken object, manipulators equipped with a gripper or vacuum cups for fixing the robot when working on submerged structures or near flat walls, manipulators equipped with inspection devices, dexterous intervention manipulators for maintenance and repair operations on submersed structures, etc. Work class ROVs are generally equipped with two manipulators. In most cases with one advanced seven function manipulator (six DOFs plus the gripper) which performs the actual task and one less advanced five function manipulator which is used to grab onto the hydro engineering structure on which the intervention operation is to be done. However, it is not unusual for an ROV to be equipped with two advanced seven function manipulators. Some of the tasks which underwater manipulators are employed for include pipe inspection, salvage of sunken objects, mine disposal, cleaning surfaces, opening and closing of valves, drilling, rope cutting, cable laying and repair, clearing debris and fishing nets, biological and geological sampling, archeological work, etc. The robot manipulators employed however on subsea ROVs are generally not servo controlled systems supported with kinematic engine control approaches. The majority of the subsea ROV manipulators are hydraulically driven devices, tele-operated with an open loop control system, completely reliant on the pilot who is located on the support vessel. The

pilot observes the scene with visual feedback through camera and / or forward looking sonar systems and simultaneously takes decisions regarding the motion of the underwater manipulator [2]. The difficulties such as poor visibility, poor 3D perception based on 2D image, make tasks that would be trivial for an industrial robot, become time consuming even for a very skilled pilot/operator and therefore very expensive. The cost of mobilising a support vessel with ROV systems can cost €18,000 for research vessels and well in excess of €50,000 for oil and gas touch down operations support per day.

Research has been done in the field of autonomous underwater manipulation by various research centers over the last twenty five years, but this has not yet been adopted in ocean engineering commercial off shore ROVs which still predominantly employ hydraulic robot arms with human in the loop control approaches with camera views of the scene. Some of the pioneering work in autonomous manipulation has been done within the OTTER [3] and AMADEUS [4] projects, while one of the state of the art research results have been achieved within the TRIDENT FP7 project [5] where an electrically powered underwater manipulator produced by Graal Tech mounted on an AUV has been used to autonomously detect and retrieve an object from the seabed [6].

Most of the autonomous manipulation research work that has been done has been developed and tested using electrical manipulator prototypes. Our goal is to enhance the capabilities of subsea manipulation by designing advanced control systems that can be implemented on off the shelf hydraulic manipulators as used on the majority of commercial ROVs with minimal adaptations.

III. Sub-Sea Manipulation Task Analysis

In this paper we analyse the scenarios for implementation of servo control approaches for marine robot manipulation on a ROV equipped with two manipulators. Such is the case with ROV Holland I – an Irish Marine Institute owned ROV used mainly for scientific purposes equipped with two seven function Schilling ORION manipulators (Fig. 1). One of the typical tasks for this type of ROV is to collect a sample from the seabed and put it in a sample container which is fixed on the vehicle or held by the other manipulator. The scenarios being investigated start with relatively straight forward tasks and advance to more challenging applications.

Scenario 1: ROV equipped with a five function manipulator and a seven function manipulator is to perform a task of placing the grasped sample in a sample container. The seven function manipulator is to be used for loading the sample into the sample container which is held by the five function manipulator. The scenario in which the sample container is fixed on the vehicle is skipped as it is the simplification of this scenario. Both manipulators are assumed to have position sensors such as resolvers in each joint. The information provided by such position sensors along with the known relative pose between the robot manipulator bases, are sufficient to develop an algorithm which provides the end effector trajectory, in either Cartesian or joint space, which lead to the task execution using standard forward and inverse kinematics techniques common in industrial robotics.

Scenario 2: The only difference between this scenario and the previous scenario is that the five function manipulator is not as advanced, i.e. it does not have position sensors (resolvers on each joint) as is common in underwater robotics. This shortfall is to be overcome by adding a vision system in the control loop. Using a camera system mounted either on the robot manipulator or the ROV, visual servoing techniques can be implemented in order to develop algorithms which generate the desired end effector motion. Both image based and position based visual servoing strategies are to be addressed.

Scenario 3: The above scenarios deal with the automation of the loading task stage not including the target object acquisition. This scenario goes towards automating the target acquisition / grasping as well and it deals with the interaction of the seven function arm with a target object independent of the ROV base platform. Both static and non-static targets are to be addressed.

The developed algorithms are to be tested in simulations as well as on experimental setup. The robot arm hardware employed in the experimental work to date includes: two 6 axis servo controlled all electric Staubli TX60 robot arms (Fig. 2), a Schilling Titan 2 hydraulic seven function subsea manipulator (Fig. 3), a Point Grey Bumblebee 2 stereo vision system (Fig. 4) and the robot control software developed by the authors using Matlab Robotics Toolbox and LabVIEW. The developed software can be employed for real robot manipulator motion control as well as in simulations using mathematical models with virtual reality animation designed with Virtual Reality Modelling Language (VRML).

Fig. 1. ROV Holland I with two Schilling Orion Manipulators

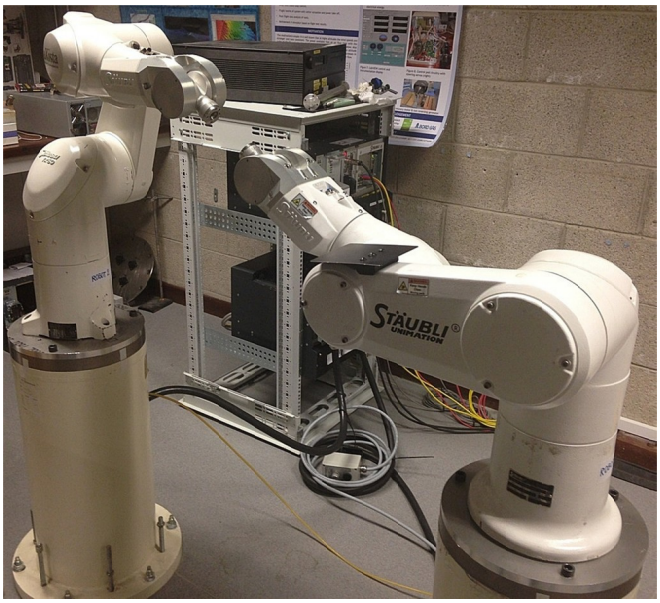

Fig. 2. Two Staubli TX60 Robot Arms

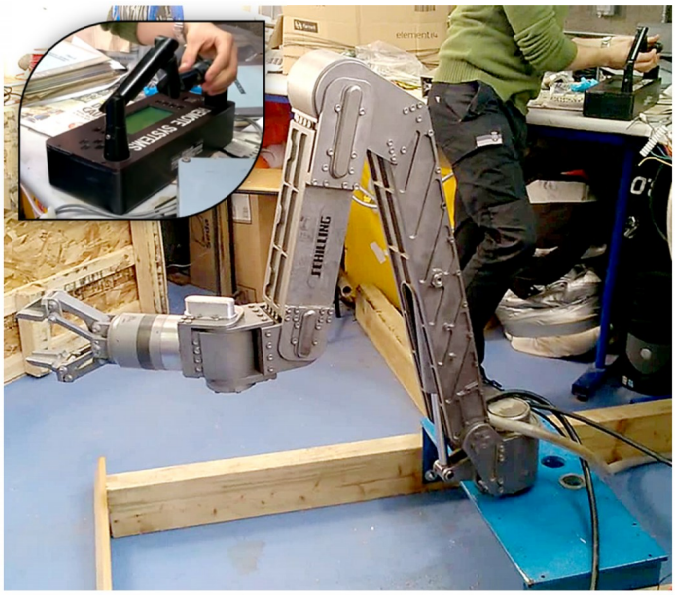

Fig. 3. Schilling Titan 2 - Hydraulic Seven Function Subsea Manipulator

Fig. 4. Point Grey Bumblebee 2 - Stereo Vision System

IV. Ongoing Development Work

In this section we will describe the designed algorithms that cover the scenario 1 which has been addressed in the previous chapter. The described software was developed using Matlab and LabView toolboxes, mainly Robotics Toolbox [1] and is designed for testing in both simulations and real experimental setup. The simulation results using two Staubli TX60 robot models as well as a Staubli TX60 and Schilling Titan 2 manipulator model combination are presented. The developed software has been tested on two real Staubli TX60 robots. The software using the real Staubli TX60 and Schilling Titan 2 robots has not been tested yet but preliminary results will be presented at the conference. The developed algorithms are purely kinematical, in the sense that they provide kinematics parameters which are to be forwarded to the existing manipulator's hydraulic servo control unit as an input as we believe that the of the shelf manipulators have sufficient capabilities to be able to cope with this inputs.

A. Kinematics Model

Both Staubli TX60 and Schilling Titan 2 manipulators are modeled as an open kinematic chain of seven rigid links connected by means of six revolute joints. Using Denavit-Hartenberg convention (DH), coordinate frames (Frame 0 to Frame 6) are assigned to each joint (Fig. 5 and Fig. 6), and DH parameters acquired (Table 1 and Table 2).

B. Forward Kinematics

Knowing DH parameters, individual coordinate transformations can be calculated using the expression

$$\boldsymbol{A}_i^{i-1}(q_i) = \begin{bmatrix} \mathrm{c}\,\vartheta_i & -\mathrm{s}\,\vartheta_i\,\mathrm{c}\,\alpha_i & \mathrm{s}\,\vartheta_i\,\mathrm{s}\,\alpha_i & a_i\,\mathrm{c}\,\vartheta_i \\ \mathrm{s}\,\vartheta_i & c\vartheta_i\,\mathrm{c}\,a_i & -c\vartheta_i\,\mathrm{s}\,a_i & a_i\,\mathrm{s}\,\vartheta_i \\ 0 & \mathrm{s}\,a_i & \mathrm{c}\,a_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (1)$$

Finally, the direct kinematics function can be constructed

$$\boldsymbol{T}_6^0(q_1, q_2 \dots q_6) = \boldsymbol{A}_1^0(q_1)\boldsymbol{A}_2^1(q_2) \dots \boldsymbol{A}_6^5(q_6). \quad (2)$$

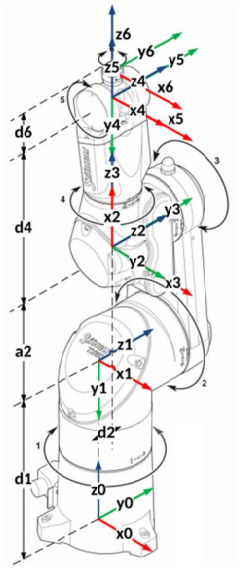

Fig. 5. Kinematics Model of Staubli TX60 Robot

	a_i	α_i	d_i	ϑ_i
1	0	−90°	0.375m	ϑ_1
2	0.29m	0	0.02m	$\vartheta_2 - 90°$
3	0	90°	0	$\vartheta_3 + 90°$
4	0	−90°	0.31m	ϑ_4
5	0	90°	0	ϑ_5
6	0	0	0.07m	ϑ_6

Table 1. Staubli TX60 DH parameters

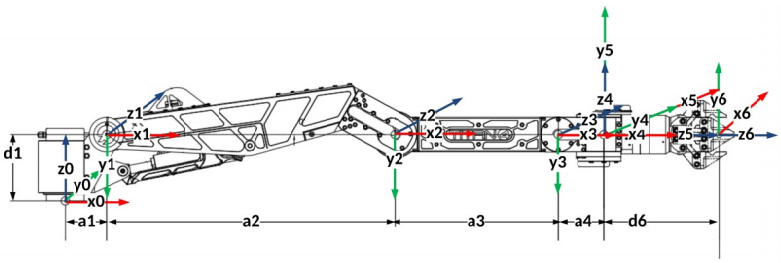

Fig. 6. Kinematics Model of Schilling Titan 2 Robot

	a_i	α_i	d_i	ϑ_i
1	0	$-90°$	0.375m	ϑ_1
2	0.29m	0	0.02m	$\vartheta_2 - 90°$
3	0	90°	0	$\vartheta_3 + 90°$
4	0	$-90°$	0.31m	ϑ_4
5	0	90°	0	ϑ_5
6	0	0	0.07m	ϑ_6

Table 2. Schilling Titan 2 DH Parameters

C. Inverse Kinematics

The problem of inverse kinematics can be defined as the determination of joint variables corresponding to a given end effector position and orientation, i.e. given a 4x4 homogenous transformation

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0} & 1 \end{bmatrix}, \tag{3}$$

find a solution, or multiple solutions if possible, of the equation

$$\boldsymbol{T}_6^0(q_1, q_2 \dots q_6) = \boldsymbol{H}. \tag{4}$$

Since TX60 robot has a spherical wrist (the last three joints axes intersect at a point) it is possible to find an analytical closed form inverse kinematics solution and this can be done by means of kinematic decoupling, i.e. separating the inverse kinematics problem into two simpler problems – inverse position kinematics and inverse orientation kinematics [7].

The Schilling Titan 2 manipulator does not have a spherical wrist and therefore finding a closed form analytical solution is impossible. Hence, the problem of solving the inverse kinematics has been done by means of numerical methods. The algorithm we have employed is the closed loop second-order inverse kinematics algorithm with pseudo inverse Jacobian [8].

The well-known differential kinematics equation which represents the relationship between the joint space velocities and the task space velocities has the following form

$$\dot{\boldsymbol{x}}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{5}$$

Derivation of this equation yields the second order differential kinematics equation

$$\ddot{\boldsymbol{x}}_e = \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}. \tag{6}$$

Inverting this equation yields a solution in terms of joint accelerations

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q})(\ddot{\boldsymbol{x}}_e - \dot{\boldsymbol{J}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}), \tag{7}$$

with the assumption that the Jacobian matrix $\boldsymbol{J}$ is non-singular and square. By integrating this equation over time, joint velocities $\dot{\boldsymbol{q}}(t)$ and positions $\boldsymbol{q}(t)$ can be reconstructed. However, open loop solutions through numerical integration methods suffer from solution drift and thus lead to operational space errors. This shortfall can be overcome by using a closed loop inverse kinematics algorithm. Utilizing this algorithm requires defining a six dimensional operational space error vector

$$\boldsymbol{e}(t) = \begin{bmatrix} \boldsymbol{e}_p(t) \\ \boldsymbol{e}_o(t) \end{bmatrix} \tag{8}$$

as the difference between the desired and actual end-effector position and orientation

$$\boldsymbol{e}(t) = \begin{bmatrix} \boldsymbol{e}_p(t) \\ \boldsymbol{e}_o(t) \end{bmatrix} = \boldsymbol{x}_d(t) - \boldsymbol{x}_e(t), \tag{9}$$

its time derivative which is effectively the difference between the desired and actual end-effector velocity

$$\dot{\boldsymbol{e}}(t) = \dot{\boldsymbol{x}}_d(t) - \dot{\boldsymbol{x}}_e(t), \tag{10}$$

and its second derivative

$$\ddot{\boldsymbol{e}}(t) = \ddot{\boldsymbol{x}}_d(t) - \ddot{\boldsymbol{x}}_e(t). \tag{11}$$

Position error $\boldsymbol{e}_p$ from (8) is calculated in a straight forward manner as the difference the between desired and actual end-effector position

$$\boldsymbol{e}_p(t) = \boldsymbol{p}_d(t) - \boldsymbol{p}_e(t) \tag{12}$$

End-effector orientation being represented by the means of unit quaternions, the orientation error $\boldsymbol{e}_o$ from (8) is calculated by the expression

$$\boldsymbol{e}_o(t) = \eta_e(\boldsymbol{q})\boldsymbol{\epsilon}_d - \eta_d\boldsymbol{\epsilon}_e(\boldsymbol{q}) - \boldsymbol{S}(\boldsymbol{\epsilon}_d)\boldsymbol{\epsilon}_e(\boldsymbol{q}), \tag{13}$$

where $Q_d = \{\eta_d, \boldsymbol{\epsilon}_d\}$ and $Q_e = \{\eta_e, \boldsymbol{\epsilon}_e\}$ are assumed to denote, respectively quaternions associated with desired and actual end-effector orientation, and $\boldsymbol{S}(\cdot)$ is a skew-symmetric operator.

The actual end-effector velocity from (10) is obtained by

$$\dot{\boldsymbol{x}}_e(t) = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{14}$$

where $\boldsymbol{J}$ represents the geometric Jacobian.

Using (6) in (11) yields

$$\ddot{\boldsymbol{e}} = \ddot{\boldsymbol{x}}_d - \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \tag{15}$$

By choosing the joint acceleration vector as

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger}(\boldsymbol{q})(\ddot{\boldsymbol{x}}_d + \boldsymbol{K}_D\dot{\boldsymbol{e}} + \boldsymbol{K}_P\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}) \tag{16}$$

we get the equivalent linear error system

$$\ddot{\boldsymbol{e}} + \boldsymbol{K}_D\dot{\boldsymbol{e}} + \boldsymbol{K}_P\boldsymbol{e} = \boldsymbol{0} \tag{17}$$

which under the assumption that $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ are positive definite matrices is asymptotically stable, i.e the error tends to zero along the trajectory. $\boldsymbol{J}^{\dagger}$ is the pseudo inverse of the geometric Jacobian $\boldsymbol{J}$ and it is used when the manipulator is redundant for a given task. This solution minimizes the norm of joint accelerations.

Integrating equation (16) over time leads to reconstruction of joint velocities $\dot{\boldsymbol{q}}(t)$ and positions $\boldsymbol{q}(t)$ assuming the initial conditions $\dot{\boldsymbol{q}}(0)$ and $\boldsymbol{q}(0)$ are known. The integration is done in discrete time using the Euler integration method. Namely, knowing the joint positions, velocities and accelerations at time t_k, joint velocities and positions at time t_{k+1} can be computed as

$$\dot{\boldsymbol{q}}(t_{k+1}) = \dot{\boldsymbol{q}}(t_k) + \ddot{\boldsymbol{q}}(t_k)\Delta t \tag{18}$$

$$\boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + \dot{\boldsymbol{q}}(t_k)\Delta t \tag{19}$$

where Δt is a given integration interval.

The described numerical closed loop inverse kinematics algorithms have been developed and implemented for both the Staubli TX60 and Schilling Titan 2 robots.

D. Simulation and experimental results for Scenario 1

This scenario 1 was simulated using two Staubli TX60 robot models. The relative pose between bases of two robots is known, i.e. it is measured from the experimental setup and expressed by means of homogenous transformation. Both robots start from initial position defined in joint space by $\boldsymbol{q} = (0,0,0,0,0,0)$. The Robot 1 then moves to the "arbitrary" but predefined position defined in joint space, simulating the action of placing the sample container in the workspace of Robot 2. Robot 2 than approaches Robot 1 with a certain offset moving on a trajectory defined in joint space, after which it moves on a straight line in Cartesian space in order to reach the point of Robot 1 end-effector simulating placing the sample into the container. Finally, Robot 2 pulls back and moves to the initial position. This has been done by using the forward kinematics equation to determine the Cartesian position of Robot 1 and transforming it in the coordinate frame of Robot 2 using homogenous transformation and then generating trajectories using the inverse kinematics algorithms described in the previous section. The joint space trajectories are constructed by means of the 5th order polynomial and Cartesian space trajectories using Trapezoidal velocity profile. Figure 7 and figure 8 represent x, y and z coordinates over time for Robot 1 and Robot 2 respectively from which it can be seen that Robot 2 reaches the desired point in the 20th second of simulation. Same simulations have been conducted with one Staubli TX60 robot and one Schilling Titan 2 robot providing similar results. Figure 9 shows the trace of trajectories after the simulation. Figures 10 and 11 presents screenshots of the virtual reality animation realized with VRML.

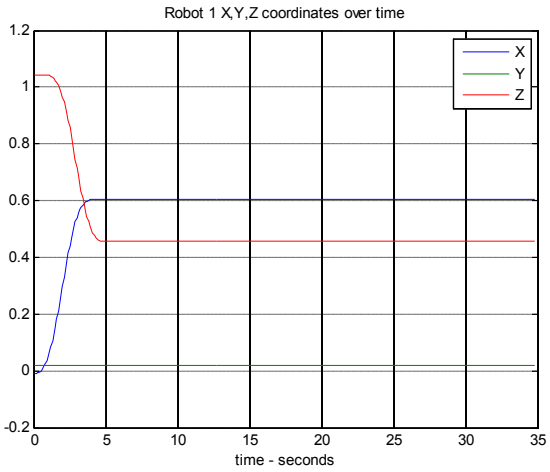

Fig. 7. Cartesian End-effector x, y and z Coordinates over Time for Robot 1 for Scenario 1 simulation

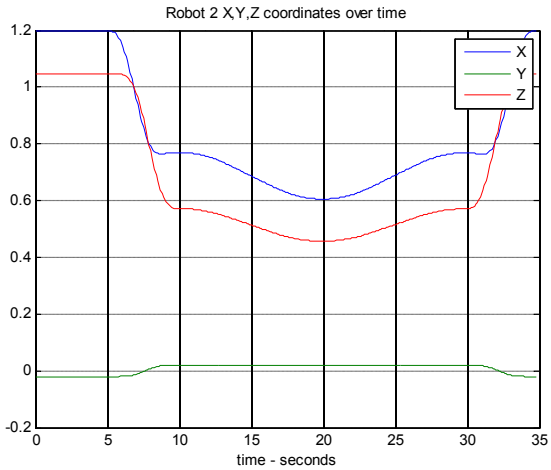

Fig. 8. Cartesian End-effector x, y and z Coordinates over Time for Robot 2 for Scenario 1 Simulation

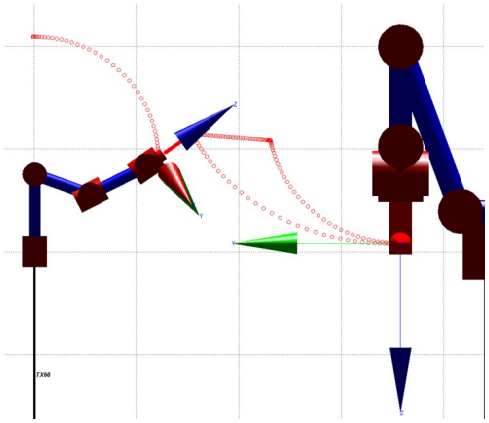

Fig. 9. Trajectory Trace of End-effector Trajectories after the Simulation

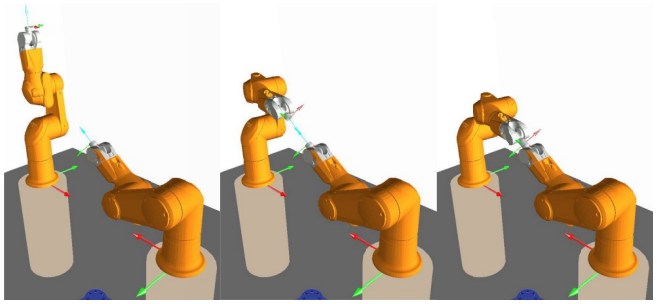

Fig. 10. Screenshots of Scenario 1 Simulation with two Staubli TX60 Robot Models

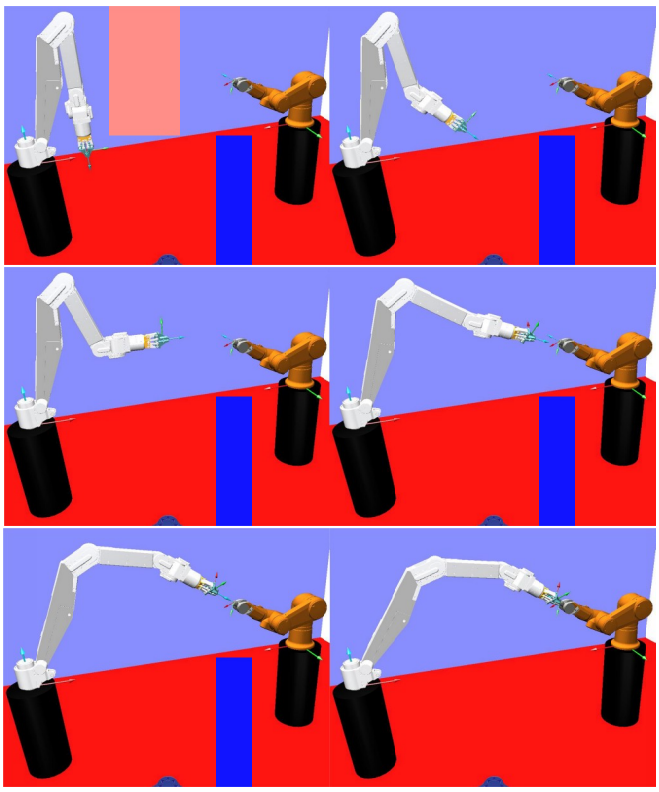

Fig. 11. Screenshots of Scenario 1 Simulation with Staubli TX60 and Schilling Titan 2 Robot Models

Real experiment was conducted utilizing the same algorithms using two Staubli robots. The main software was running on the PC communicating over the Ethernet with two Staubli TX60 CS8C controllers. The standard programming language for Staubli Robots called VAL3 was used to preprogram the two robots for the experiment. Robot 1 is preprogramed to move to the position defined by the user at the start of the experiment after which it continuously sends its position in joint space over the Ethernet to the PC. The other robot is preprogramed to continuously move to the position in Cartesian space received over the Ethernet from the PC. Figure 12 shows photos taken during the experiment. For safety reasons a certain offset was used which is why the Robot 2 end-effector does not completely reach the Robot 1 end-effector which can be seen on the most right image of Fig. 12.

A LabVIEW aapplication has been developed for the purpose of testing the described algorithms. This application can be used for both the Staubli TX60 robot and Schilling Titan 2 robot. Figure 13 shows the user interface of the developed application. In terms of joint space, the user can manually control each Robot's joint separately by moving sliders or set a final position in joint space (six angles) and can generate a smooth trajectory from the initial position to the desired position. In terms of Cartesian space, the user can manually move the Robot's end-effector in three directions (x, y, and z) in the end-effector coordinate frame or set the desired point in Cartesian space and generate a trajectory from the initial to the desired position forcing the robot's end effector to move in the straight line. As in simulations, a Cartesian trajectory is constructed using a trapezoidal velocity profile. The desired position is defined with three coordinates in the base coordinate frame with options to keep orientation the same as in the initial position or to choose the desired orientation using either roll pitch yaw or angle axis representation. The user interface also provides real time animation of robot motion. No matter in what manner the operator generates the robot motion, the outputs of the software are joint angles and velocities. These values are to be mapped to the inputs of the Commercial Schilling Titan 2 controller in order to test the software on this real robot. The preliminary results of the experiments are to be presented at the conference.

V. Conclusions And Future Work

By using algorithms common for industrial robotics we successfully developed a kinematics engine which enables generating reference kinematics parameters for motion both in joint and Cartesian space. Using the developed software we believe that ROV pilots would be able to execute typical subsea manipulation task easier and faster.

The next step will be investigating and developing visual servoing algorithms and encapsulating them into our software. By doing that we would progress towards solving problems of Scenarios 2 and 3 described in chapter III. Both image based and position based algorithms are to be investigated.

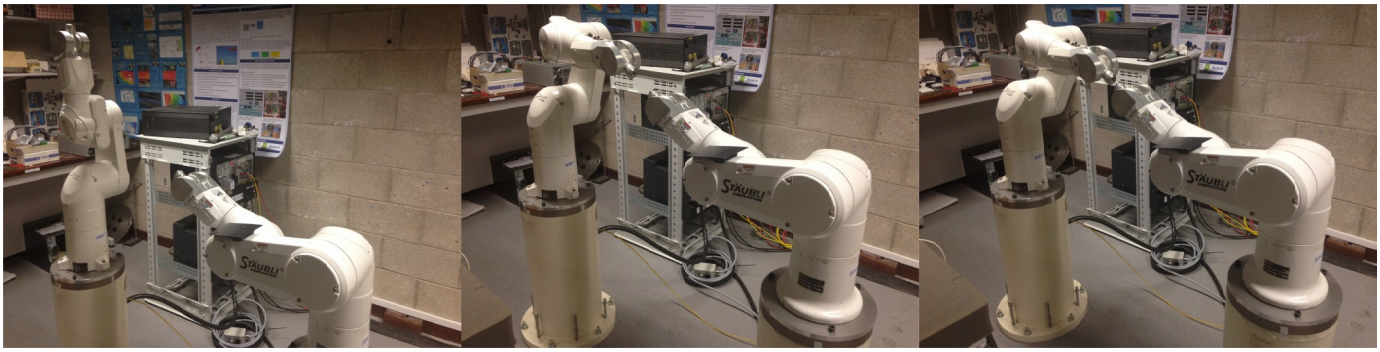

Fig. 12. Photos taken during the Scenario 1 Experiment

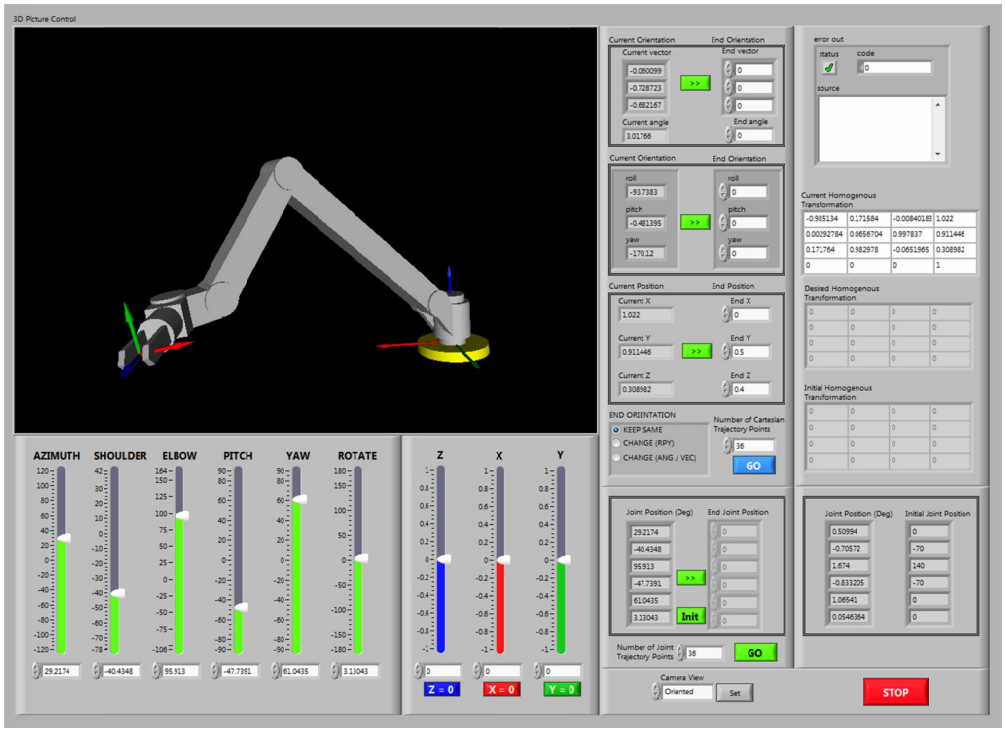

Fig. 13. Developed LabVIEW Application Front Panel of Virtual Instrument

ACKNOWLEDGMENT

This publication has emanated from research supported by the Science Foundation Ireland under the MaREI Centre research programme [Grant No. SFI/12/RC/2302]. The MaREI project is also supported by the following industrial partners: Resolve Marine, Shannon Foynes Port Company, Teledyne Blueview and The Commissioners of Irish Lights.

REFERENCES

[1] P. Corke, *Robotics, Vision and Control*, vol. 73. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[2] J. Yuh, "Underwater robotics," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, pp. 932–937.

[3] H. H. Wang, S. M. Rock, and M. J. Lees, "Experiments in automatic retrieval of underwater objects with an AUV," in *"Challenges of Our Changing Global Environment". Conference Proceedings. OCEANS '95 MTS/IEEE*, 1995, vol. 1, pp. 366–373.

[4] D. Lane, J. Davies, G. Casalino, G. Bartolini, G. Cannata, G. Veruggio, M. Canals, C. Smith, D. J. O'Brien, M. Pickett, G. Robinson, D. Jones, E. Scott, A. Ferrara, D. Angelleti, M. Coccoli, R. Bono, P. Virgili, R. Pallas, and E. Gracia, "AMADEUS: advanced manipulation for deep underwater sampling," *IEEE Robot. Autom. Mag.*, vol. 4, no. 4, pp. 34–45, 1997.

[5] P. Cieslak, P. Ridao, and M. Giergiel, "Autonomous underwater panel operation by GIRONA500 UVMS: A practical approach to autonomous underwater manipulation," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, vol. 2, no. 1, pp. 529–536.

[6] D. Ribas, P. Ridao, A. Turetta, C. Melchiorri, G. Palli, J. J. Fernandez, and P. J. Sanz, "I-AUV Mechatronics Integration for the TRIDENT FP7 Project," *IEEE/ASME Trans. Mechatronics*, pp. 1–10, 2015.

[7] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Wiley New York, 2006.

[8] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics*. London: Springer London, 2009.