Cooperative Assembly for Mobile Manipulators in an Underwater Mission Scenario

Davide Torielli

DIBRIS - Department of Computer Science, Bioengineering, Robotics and System Engineering

University of Genova

In partial fulfillment of the requirements for the degree of

Robotics Engineering

Semptember, 2019

Contents

Chapter 1

Introduction

Nowadays, different kinds of robot are largely used for underwater missions. A particular type of submarine robot is the Underwater Vehicle Manipulator System (UVMS): an autonomous underwater vehicle (AUV) capable of accomplishing tasks that require a certain level of dexterity, thanks to single or multiple arms. A really innovative field for underwater missions is the analysis of cooperation between multiple agents, which permits to extend their flexibility. Cooperative robots are two or more robots, identical or different, that coordinate themselves autonomously (i.e. without human intervention) to accomplish various objectives, from mapping an area to assembling an object.

This thesis focuses on a totally unexplored environment: cooperative *peg-in-hole* assembly with underwater manipulators. It is part of the TWINBOT project [TWINBOT (2019)], which is devoted to make a step forward to missions in complex scenarios.
The experimental set-up is made up of two I-AUV's Girona 500 AUV underwater vehicles, each one equipped with a CSIP Robot arm5E (4 DOF arm with a parallel yaw gripper). The final goal is to successfully coordinate the robots in such a way that the peg, hold by one manipulator, is inserted correctly in the other piece, hold by the second manipulator. The work focuses on the kinematic layer, adopting inverse kinematic algorithms based on task's priority (TPIK), although for the peg-in-hole task particular attention must be paid to dynamics interactions. Also the communication problems that arise under the water must be taken into account.
The whole work is divided in two step:

The first step is to adapt solutions of previous works to pick up, recover and transport the objects using two intervention vehicles. This step is based on previous results from MARIS [Casalino *et al.* (2014)], TRIDENT [Ribas *et al.* (2015)], and MERBOTS [Centelles *et al.* (2017)] projects.

The step further analyses the peg-in-hole problem, with the aim to solve the

assembly task (*M4* milestone of TWINBOT). The chosen strategy divides the problem in two phases: Approaching and Insertion. In the first preliminary tasks are done to prepare the assembly, like aligning correctly the end-effectors of the two manipulators. The second phase explores all the problems inherent to the interaction between the *peg* and the *hole*. The work aims to find a solution to make the robots *cooperate*, understanding the forces exchanged between them while they are assembling the parts.

1.1 State of the Art

Robots have been massively introduced in various fields to help humans in different tasks. Nowadays, there are strategies to use them in underwater environments. A manipulator (robot arm) is considered to be the most suitable tool for executing sub-sea intervention operations. Hence, unmanned underwater vehicles (UUVs), such as remotely operated vehicles (ROVs) and autonomous underwater vehicles (AUVs), are equipped with one or more underwater manipulators.

During the last 20 years, underwater manipulators have been used for many different sub-sea tasks in various fields, for example, underwater archaeology [Bingham *et al.* (2010); Drap (2012)], marine geology [Wynn *et al.* (2014); Urabe *et al.* (2015)], and military applications [Capocci *et al.* (2017)].

There are many specific tasks where the underwater manipulators are important, from salvage of sunken objects [Cheng Chang *et al.* (2004)] to mine disposal [Fletcher (2000); Djapic *et al.* (2013)]. One particular scenario is towards the oil and gas industry, where underwater manipulators are used for pipe inspection, opening and closing valves, drilling, rope cutting [Christ & Wernli (2013)], and, in general, to reduced field maintenance and development costs [Gilmour *et al.* (2012)]. A recent survey explored the market related to this technology for oil and gas industry [Diaz Ledezma *et al.* (2015)].

1.1.1 Previous Works in Underwater Missions

Since early '90s, research in marine robotics started focusing on the development of underwater vehicle manipulator systems (UVMS). ROVs have been largely used, but they have high operational costs. This is due to the need for expensive support vessels, and highly qualified man power effort. In addition, the pilot which operates the vehicle and the arm experiences heavy fatigue in order to carry out the manipulation task.

For the above reasons, the research started to increase the effort toward augmenting the autonomy level in underwater manipulation. For example, to reduce

the operators fatigue, some autonomous control features were implemented in work class ROVs [Schempf & Yoerger (1992)]. Another solution is to completely replace ROVs with autonomous underwater vehicles (AUVs).

Some pioneering projects in this field carried out in the '90s. The AMADEUS project [Lane *et al.* (1997)] developed grippers for underwater manipulation [Casalino *et al.* (2002)] and studied the problem of dual arm manipulation [Casalino *et al.* (2001)].

The UNION project [Rigaud *et al.* (1998)] was the first to perform a mechatronic assembly of an autonomous UVMS.

Early 2000s showed many field demonstrations. The SWIMMER project [Evans *et al.* (2001)] developed a prototype autonomous vehicle to deploy a ROV mounted on an AUV. This permitted to remove the need of long umbilical cables and continuous support by vessels on sea surface.

This work was followed by the ALIVE project [Evans *et al.* (2003); Marty (2004)], that achieved autonomous docking of Intervention-AUV (I-AUV) into to a sub-sea structure not specifically created for AUV use.

The SAUVIM [Yuh *et al.* (1998); Marani *et al.* (2009)] project carried out the first autonomous floating underwater intervention. It focused on the searching and recovering of an object whose position was roughly known a priori. Here, the AUV weighted 6 tons, and the arm only 65 kg, so the dynamics of the two subsystems were practically decoupled and the two controllers were separated. The mission consisted in the AUV performing station keeping while the arm was recovering the object.

After SAUVIM, a project called RAUVI [Prats *et al.* (2012)] took a step further. Here, the AUV performed a hook-based recovery in a water tank. Again, the control of the vehicle and the arm was separated, even if the Girona 500 light AUV and the small 4-degrees-of-freedom (DOF) arm used had more similar masses than the ones used in SAUVIM.

A milestone was the TRIDENT project [Sanz *et al.* (2012)]. For the first time, the vehicle and the arm were controlled in a coordinated manner [Casalino *et al.* (2012)] to recovery a black-box mockup [Simetti *et al.* (2014a)]. The used task priority solution dealt with both equality and inequality control objectives, although the inequality ones were only-scalar, except for the joint limits. This permitted to perform some manipulation tasks *while* considering also other objectives, for example, keeping the object centred in the camera frame. In this project, only the kinematic control layer (and not also a suitable dynamic one) was implemented.

The PANDORA project [Lane *et al.* (2012)] focused on increasing the autonomy of the robot, by recognizing failures and responding to them. The work combined machine learning techniques [Carrera *et al.* (2014)] and a task priority

kinematic control approach [Cieslak *et al.* (2015)]. However it dealt with only equality control objectives, with a specific ad-hoc solution to manage the joint limit inequality task.

The TRIDENT concepts were enhanced within the MARIS project [Casalino *et al.* (2014)]. The used task priority framework [Simetti & Casalino (2016)] permitted to *activate* and *deactivate* equality/inequality control objectives of any dimension (not only scalar ones). This project also extended the problem to cooperative agents [Simetti & Casalino (2017)].

TRIDENT and MARIS concentrated on using the control framework to perform only grasping actions. Recent work [Simetti *et al.* (2018)] analyses how the method can be used in different scenarios, like pipeline inspection and deep sea mining exploration.

The DexROV project [Di Lillo *et al.* (2016)] is studying latencies problems which arise de-localizing on the shore the manned support to ROV operations.

The PROMETEO project [PROMETEO (2016)] plans to improve the use of underwater robotics in archaeological sites. It investigates the manipulation capacity when occlusions of objects can occur and with a wireless communication system to use the robot without umbilical cable.

The ROBUST project [ROBUST (2016)] aims to explore and to map deep water mining sites, through the fusion of two technologies: laser-based in-situ element-analysing, and AUV techniques for sea bed 3D mapping.

1.1.2 The Control Framework

In the '90s, industrial robotics researches focused on how to specify control objectives of a robotic system. This was done especially for redundant systems, i.e. systems with more degree of freedoms (DOFs) than necessary. This surplus is useful to perform multiple, parallel tasks; for example, avoiding an obstacle with the whole arm while the end-effector is reaching a goal. Given that such systems need to complete different goals, it has become important to have a simple and effective way to specify the control objectives.

The task-based control [Nakamura & Hanafusa (1986)], also known as operational space control [Khatib (1987)], defined the control objectives in a coordinate system that is directly linked to the tasks that need to be performed. This idea was followed by the concept of task priority [Nakamura (1990)]. In this theory, a more important task is executed together with a less important task. To accomplish the whole action, the secondary task is attempted only in the null space of the primary one. This means that the secondary task is executed *only if* it does not go against the accomplishment of the first.

This concept was later generalized to multiple task priority levels [Siciliano & Slotine (1991)]. These works putted the position control of the end-effector as the highest prioritized one, while safety tasks (like joint limits) were only *attempted* at lower priority level.

First studies in control of redundant manipulators [Yoshikawa (1984); Maciejewski & Klein (1985)] managed the free residual DOF in such a way to solve the problem of singularity and obstacle avoidance for an industrial manipulator. Another work [Khatib (1986)] introduced the use of potential functions in industrial manipulators and mobile robots.

A different solution [Chiaverini (1997)] proposed a suboptimal approach. The secondary task was solved as if it was alone, but after it was projected in the null space of the higher priority one. To deal with singularities, a variable damping factor was used [Nakamura & Hanafusa (1986)]. This solution was later enhanced and called null-space-based behavioural control [Antonelli *et al.* (2008)].
The approach does not deal with the problem of algorithmic singularities that can occur due to rank loss caused by the projection matrix. Further works [Marani *et al.* (2003); Flacco *et al.* (2012)] focused on this problem.

Since those times, the task priority framework has been applied to numerous robotic systems, other than redundant industrial manipulators. Some examples includes mobile manipulators [Antonelli & Chiaverini (1998); Antonelli & Chiaverini (2003); Zereik *et al.* (2011)], multiple coordinated manipulators [Padir (2005); Simetti *et al.* (2009)], modular robots [Casalino *et al.* (2009)], and humanoid robots [Sentis & Khatib (2005); Sugiura *et al.* (2007)]. Furthermore, a stability analysis for several prioritized inverse kinematics algorithms can be found in [Antonelli (2009)].

The problem of the classical task priority framework, evident in all the previous mentioned works, is that inequality control objectives (e.g. avoiding joint limits) were never treated as such. In fact, the corresponding tasks were always active, like the equality ones. So, for example, also when the joints are sufficiently far from their limits, the fact that the task is active uselessly adds constraints and "consumes" DOFs. Thus, without a transition, the safety control objectives like joint limits could be only considered as secondary. Otherwise, they would consume DOFs, and mission tasks, like reaching a position with the end-effector, can never be accomplished. This led to an undesired situation where safety tasks have a lower priority with respect to non-safety ones.

The challenge in activating (inserting) or deactivating (deleting) a task is that these transitions would imply a discontinuity in the null space projector, which

leads to a discontinuity in the control law [Lee *et al.* (2012)]. Thus, in the last decade, researches focused on integrate safety inequality control objectives in a more efficient way.

A new inversion operator was introduced [Mansard *et al.* (2009b)] for the computation of a smooth inverse with the ability of enabling and disabling tasks in the context of visual servoing. But the work only dealt with the activation and deactivation of the rows of a single multidimensional task (so, not including the concept of different levels of priority). The extension to the case of a hierarchy of tasks with different priorities was provided successively [Mansard *et al.* (2009a)]. However, the algorithm requires the computation of all the combinations of possible active and inactive tasks, which grows exponentially as the number of tasks increases.

Another work [Lee *et al.* (2012)] modified the reference of each task that was being inserted or being removed, in order to comply with the already present ones, and in such a way to smooth out any discontinuity. However, the algorithm requires $m!$ pseudo-inverses with m number of tasks. For this reason, the authors provided approximate solutions, which are suboptimal whenever more than one task is being activated or deactivated.

Another approach [Faverjon & Tournassoud (1987)] directly incorporated the inequality control objectives as inequality constraints in a Quadratic Program (QP). According to this, the idea was generalized to any number of priority levels [Kanoun *et al.* (2011)]. At each priority level, the algorithm solves a QP problem, finding the optimal solution (in a least-squares sense). Slack variables are used to incorporate inequality constraints in the minimization process. If the solution contains a slack variable different from zero, it will mean that the corresponding inequality constraint is not satisfied. Otherwise, the inequality constraints are propagated to the next level and transformed into an equality ones (to prevent lower priority tasks from changing the best least-square trade-off found). A similar process is done for the equality constraints. A drawbacks of this approach is that the cascade of QP problems can grow in dimension. Another issue is that the activation and deactivation of tasks are not considered. This last point is important when temporal sequences of tasks are used, for example when assembling objects [Nenchev & Sotirov (1994); Baerlocher & Boulic (2004)].

Instead of a cascade of QP problems, another research [Escande *et al.* (2014)] proposed to solve a single problem finding the active set of all the constraints at the same time. Due to its iterative nature, the authors proposed to limit the number of iterations to achieve a boundary on the computation time, to be more suitable for a real-time implementation. But this solution is not optimal, and, again, activation/deactivation of equality tasks is not considered.

Improvements are made in the already cited TRIDENT project [Sanz *et al.* (2012); Simetti *et al.* (2014a)], where field trials proved how to consider activation

and deactivation of scalar tasks. But the solution still lacks the ability to deal with activation/deactivation of multidimensional tasks, i.e. multiple scalar tasks at the same priority level.

The goal reached by TRIDENT are improved in the MARIS project [Casalino *et al.* (2014); Simetti *et al.* (2014b)], where, among the others accomplishment, task transitions were successfully implemented in the framework. In particular [Simetti & Casalino (2016)], possible discontinuities that can arise are eliminated by a task-oriented regularization and a singular value oriented regularization. Plus, the original simplicity of the task priority framework is retained thanks to pseudo-inverses.

1.1.3 The Peg-in-Hole Assembly Problem

The peg-in-hole is an essential task in assembly processes in various fields, such as manufacturing lines.

This task can be performed following the classical position control method. But this is possible only if precise position of the hole is provided, and the position control error of the robot is zero. In practice, these conditions can only be obtained in specialized scenario. In the case of more versatile robots, such as industrial or underwater manipulators, imprecisions and errors are unavoidable.

To deal with these problems, classical works exploit two kind of instruments: cameras and sensors. With camera(s), the robot can roughly recognize the objective (i.e. the hole) and inspect the overall process. Past researches [Miura & Ikeuchi (1998); Pauli *et al.* (2001)] use this idea to extract boundaries of the object. Another one [Chang *et al.* (2011)] uses visual feedback for a micro-peg-in-hole task (hole of $100\mu m$).

Other approaches perform precise assembly of the parts thanks to force/torque sensors installed on the wrist. A study [Shirinzadeh *et al.* (2011)] successfully accomplishes the assembly detecting the force of contact to compensate the positional uncertainty. Newman *et al.* study [Newman *et al.* (2001)] shows how sensors can be used to build map of force and torque values of each contact point. In another works [Dietrich *et al.* (2010); Abdullah *et al.* (2015)], the location of the hole was estimated using the measured reaction moment occurred by the contact. Another good aspect of the sensors [Oh & Oh (2015); Song *et al.* (2016)] is that they can guarantee stable contact through real-time contact force feedback.

Other proposals [Chhatpar & Branicky (2001); Lee & Park (2014)] try to estimate the state of the contact using joint position sensor. This permits to not use the force/torque sensors on the wrist, which would need highly control frequency, and would increase overall cost and operation time. Some researchers [Park *et al.* (2013)] show that assembly task can be accomplished without contact

force information and with inaccurate vision data. The proposed strategy mimics the human behaviour: the peg was rubbed in a point close to the object until the relevant objects mated using compliant characteristics. The compliance allows the robot to softly adapt to the hole [Lozano-Prez *et al.* (1984); Xu (2015)]. A Similar, unexpensive, approach is tested experimentally [Park *et al.* (2017)], without the use of force/torque sensors (i.e. no force feedback), nor Remote-center-compliance devices, and with inaccurate hole information.

Chapter 2

Vision

Summary

Before the twin robots can approach the hole, its position must be know, at least roughly. In this chapter, the pose estimation of the hole is discussed.
In the considered scenario, a third robot is present. Its duty is exclusively to *detect* & to *track* the hole. In the simulation, another Girona 500 AUV is used for this job, without the arm. Is evident that, in real scenario, a littler and more efficient robot should be used for the vision, see that no manipulation task are needed. In fact, in the original TWINBOT [TWINBOT (2019)] simulation, a smaller BlueROV is present, as can be seen in (TODO) However, in this case, another Girona 500 is used to not deal with another robot model.
The *vision* robot is equipped with two identical cameras which point in front of it. They are used as:

- Two distinct cameras, independent one of the other.
- As stereo cameras, thus exploiting stereo vision algorithms.
- As RGB-D camera, i.e., a stereo vision couple where the left one is a RGB camera and the right one a Depth camera.

For the sake of simplicity, some assumptions are made:

- Known *intrinsic* camera parameters. These parameters are used by algorithms to take into account how the single camera see the scene. The (known) distortion is zero.
- Known *extrinsic* camera parameters, i.e. the position and orientation of cameras (respect the vehicle), and thus the relative pose (the transformation matrix) from one camera to the other (needed for stereo vision algorithm).

- No external disturbances for the images, such as light reflections underwater or bad visibility.
- Hole model known. This means that dimensions of the cuboid which contain the hole are known. Further explanation about this are given successively in (TODO)
- A "friendly" cuboid hole. The front face is coloured and additional hole are present, as can be seen in (TODO) . This help both the *detection* phase and the *tracking* phase.

About the robot, other assumption are:

- the pose of the vehicle respect the inertial frame is known. (TODO?AS explained?? se si linka sez). This permits to know the hole pose estimation respect the inertial frame, to directly send the robot which are carrying the peg.
- The initial position of the robot is such that it is facing the front face of the hole. It must be noticed that methods explained in the next sections can be adapted to relax this hypothesis. For example, the robot could turn around z-axis until the hole is detected. (TODO?? Also, good results are obtained when the robot not exactly face directly the cuboid, but, for example, it is on its side, looking at front face and a side one.
- Once the robot has detected the hole and the pose sent to the twin robots, it must go away to not interfere with the insertion phase. This is done through keyboard (as a ROV) but it is not difficult to improve the code to let him go away autonomously. It must be noticed that, thanks to the *tracking* algorithms, if the robot moves (because it is commanded to do so, or for water currents) the pose estimation is still good.

The job is done into two phases: *Detection* 2.1 and *Tracking* 2.2

2.1 Detection

Object Detection means detect a particular shape (the *object*) in the scene. This is important to initialize the tracking algorithm. In fact, to track an object, its initial pose (or at least some particular points of it) must be know. Thus, the detection part is the most difficult part.
For the tracking algorithm used successively, the detection part must provide a correspondence between some points in the 2D image and some points in the 3D object shape. It is important to notice that the 3D coordinate needed refers to

object frame, and not to an "external" frame. Seen that the object model is assumed to be know, the 3D coordinate of some point directly derive from this, so no other assumptions are made. So, in the experiment, a .init file looks like this:

```
4               # Number of points
                #xyz with x going in, y on the right, z down. Unit measure is meter
0 -0.4 -0.4   # top right corner
0 0.4 -0.4    # top left
0 0.4 0.4     # bottom left
0 -0.4 0.4    # bottom right
```

Indicating the position on the 4 corners of the front face, respect to a frame positioned in the center of the hole, with x-axis going inside the hole, y lying along the surface pointing on the right, z pointing down to the seafloor.
Four points is the minimum number of point accepted by the tracking algorithm. The more the point are, the more the tracking is good. Plus, point should lying on different surfaces of the object, to have better results. Anyway, good tracking result are obtained also not considering these two aspects.

The work of Detection is to provide 2D coordinate as (x, y) of these point in the 2D image captured by camera. This must be done for each camera, except for the Depth one, when used.
Two method are used: *Find Square Method* and *Template Matching*. A third method, in which the 2D positions of the points are chosen by human operator, is used to have a benchmark of the other two and to analyse the tracking results when 2D Coordinates are almost perfect.
Other methods and functions are briefly explained in Appendix A

2.1.1 Already known Coordinate Method

As explained, with this method the 2D coordinate are perfectly known. The code do this by letting the user to click on the 4 pixel which contains the corners. It is assumed that the user click on the best pixels which contain the corners. Given that the image is made by discrete pixels, is impossible to have an ideal point which is exactly the corner, but the errors for this are not noticeable.

2.1.2 Find Square Method

This method is taken from an OpenCV tutorial.
Details of how each function works and explanation of the computer vision algorithm used are not provided here, to not go outside the scope of the thesis. Only a rough explanation of how the method works is presented:

- This function looks in each image channel (unique if is a gray image, three if is a colored image) to find squares.
- First, it pre-process the image to reduce noise.
- Then, *findContours()* is called to retrieves contours of the square with the algorithm described in Suzuki & Be (1985).
- Each contour is approximated to be more like regular polygon, with lesser vertices and edges.
- Finally, it look if the shapes are similar to squares/rectangles. This is done checking if the internal angles are almost 90 degree.
- The returned squares are described by its four corners, that is what we are looking for. An additional function is called to be sure that the order of the returned corner is the same order of the point described in the .init file, otherwise correspondence are obviously erroneous.

In the way it works, it should be noticeable that this algorithm give good results only if the camera approximately face the cuboid structure at the front. (TODO as can be see se ti metti di lato...). If the side face is more visible, it will be the one detected. So, we must know which side the robot is facing to give the 3D correspondence points in the .init file.
In addition, this method is suitable if no other squares of similar dimension are present, otherwise further work is needed to discriminate them. Also, is not suitable with other kind of shapes (a pipe hole, for example).
Given the described initial pose, TODO result are good.

2.1.3 Template Matching Method

Differently from the previous one, this method belong to a wider class of well-known method. *Template Matching* means to find a pattern (in this case, the face of the hole) inside a scene. So, an additional image of the square face of the hole is needed.
The code developed follow an OpenCV tutorial. As the previous method, details on how template matching works go outside the scope of this thesis.

In brief, template matching find the point in the scene which as more similarity (or less dissimilarity) with the provided template. This is done considering intensity values of the pixels in the neighbourhood area of a center pixel. In practice, the template is shifted all over the scene image and some calculation for each new template shifting are done. Various formula are provided by OpenCV and are detailed in the previous linked of the tutorial.
It is important that the template is being scaled up and down, because usually its size is not equal to the size of the object in the scene. For each scaling, a best similarity point is detected. Then, all the similarity points are compared and the best one chosen. At the end, a rectangle with the template dimension (scaled) is build considering this point as the center. The corners of the rectangle are the 4 point which we was looking for.

TODO RESULT OF THISSSSSS

This method is less robust than the previous for different initial position of the robot. If the face is view from a different angle, other template image is needed, with an orientation similar to what the robot is seeing. In general, lot of template images at different angles are needed. Also, building the shape around the centre point is more difficult, if this shape is not a square.

2.2 Tracking

Appendix A

Other Algorithms for Object Detection

During the simulations, several trials have been done to find a suitable algorithm for the detection of the hole structure. In section (TODO) two methods have been discussed as the successful ones. In this appendix, others are briefly explained and discussed. Even if they are not used in the last versions of experiments, they can be useful for other purposes, such as detection of other kind of shapes. They can also be useful when the scene is different, for example while taking the hole structure from its side.
Each one is taken from OpenCV Detection tutorials, where also other algorithms can be found.

A.1 Corner Detection with the Shi-Harris method

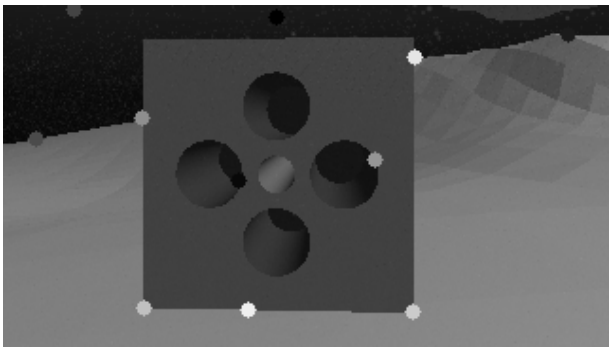

Figure A.1: *goodFeaturesToTrack()* result. Only two detected points are real corners, and the upper corners are not detected

Following the tutorial, I implemented a corner detector with the Shi-Harris method [Shi & Tomasi (2000)] using the OpenCV function goodFeaturesToTrack().
This function acts as corner detector. In our case, this can be useful to find corners of the square hole structure. These points can be used successively as starting point to higher level vision algorithms (e.g. draw the square shape to then understand its pose).
The original example lets change the number of maximum points to be found. This is useful to reduce the number of false positive corners. The main problem is that the real corners of the square are not the "best" ones. So we can't simply put this parameter equal to 4. On the other side, with bigger number of points, is then difficult to discriminate the right corners from the others.
Other interesting parameters are:

- **minDistance** The minimum distance between corners to be found.
- **qualityLevel** Parameter characterizing the minimal accepted quality of image corners.
- **blockSize** Size of an average block for computing a derivative covariation matrix over each pixel neighbourhood.
- **mask** To specify a certain region of interest in the image. In such a way, corners are found only in this region. The problem in our case is that without prior works we can't know where is the hole surface.

The points detected are effectively good feature points (as can be see in A.1). But the best ones, are not the ones that we want to detect (ie, the corner of the square).
This method should be used as a low level algorithm, to then help higher level ones. For example, to construct some polygons and to check if these polygons are square/rectangles. However, to follow this direction should be better to start from the edges. Another function can be to initialize a tracking by keypoints.

A.2 Canny Edge and Hough Transform

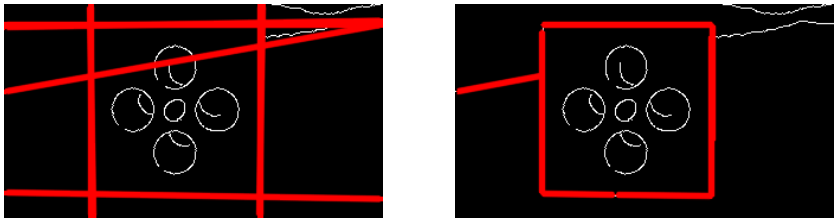

Figure A.2: Result of Hough Standard Transform (left) and Probabilistic (right). In white all the edges detected with Canny, in red the detected straight lines

The Hough Transform [Duda & Hart (1972)] is a method to detect straight lines in an image. Usually, a preprocessing of the image with an edge detector is used to improve the results, for example with a Canny Edge Detector [Canny (1986)]. The OpenCV tutorial makes use of two types of Hough Transform: the standard *HoughLines()* and the probabilistic *HoughLinesP()* [Matas *et al.* (2000)].
Results are visible in A.2. The probabilistic method shows that this function is good to detect the square structure of the hole. Thus, this method can be used as good starting point to further process the image, to then estimate the pose of the hole, or of other objects of interest.

A.3 Bounding Boxes Detection

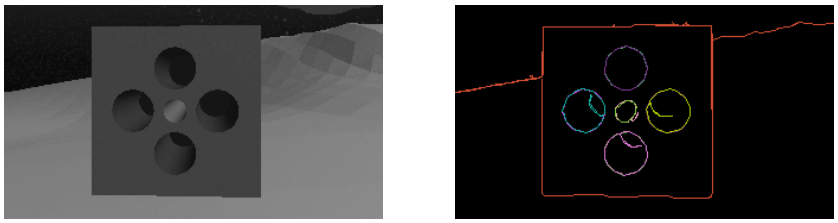

Figure A.3: Result of the algorithm: on the left the original image, on the right the contours detected

In this method, shape contours are found and then bounding boxes drawn. As explained in the previous section, finding shape contours can be a starting point to initialize higher level image processing, such as the tracking.
The code derived from an OpenCV tutorial.
First, a Canny edge detector is used to preprocess the image. Then, the function *findContours()* is called to retrieves contours with the algorithm described

in Suzuki & Be (1985). Finally, rectangles are draw around as a bounding boxes. In case of the square hole structure (already almost a "bounding box" of itself), finding the boxes is useful to reduce the noise, and to have a square/rectangle with straight lines.
The result without the bounding boxes are shown in A.3. The square is noticeable, but also other not interesting lines are shown. For this specific purpose (detection of square face of hole), the detection is worse than the algorithm of section A.2 which used probabilistic hough transform.

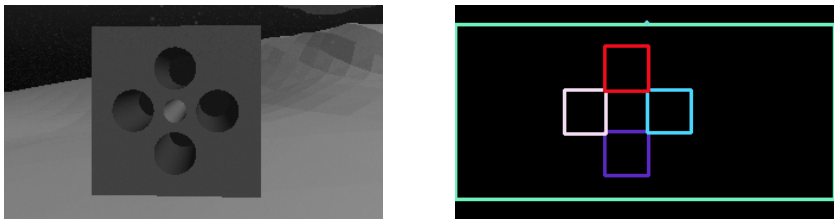

Figure A.4: On the left, the original image. On the right, the drawn bounding boxes around the holes

The thing to notice is that the holes on the surface are well visible. This may be useful for other algorithms, or to detect other kind of shapes like a tube hole (a pipe). Results in A.4 show bounding boxes around the holes. However, it is difficult to have a precise pose estimation with bounding boxes.
In conclusion, this is a good method that can be explored. However, lot of non interesting edge are detected, so parameters have to be chosen wisely.

A.4 2D Feature Matching & Homography

Image features are small patches that are useful to compute similarities between images. Please note that these features are different from corner points. They indicates particular details that are different from others. Detecting these areas is important to recognize objects of interest in the image. The *descriptors* of these features contain the visual description of the patches, and they are used to recognize similarities.
This method uses a *object image* and a *scene image*. The first is a sort of template (note that this method is not a template matching (TODO CITA MY SEZ)) which contains only the object to be found (in this case, the square face of the hole). The second is the image in which we want to detect this object (in this case, what the camera is seeing).
After good *features* are extracted from both images, the *descriptors* are used to

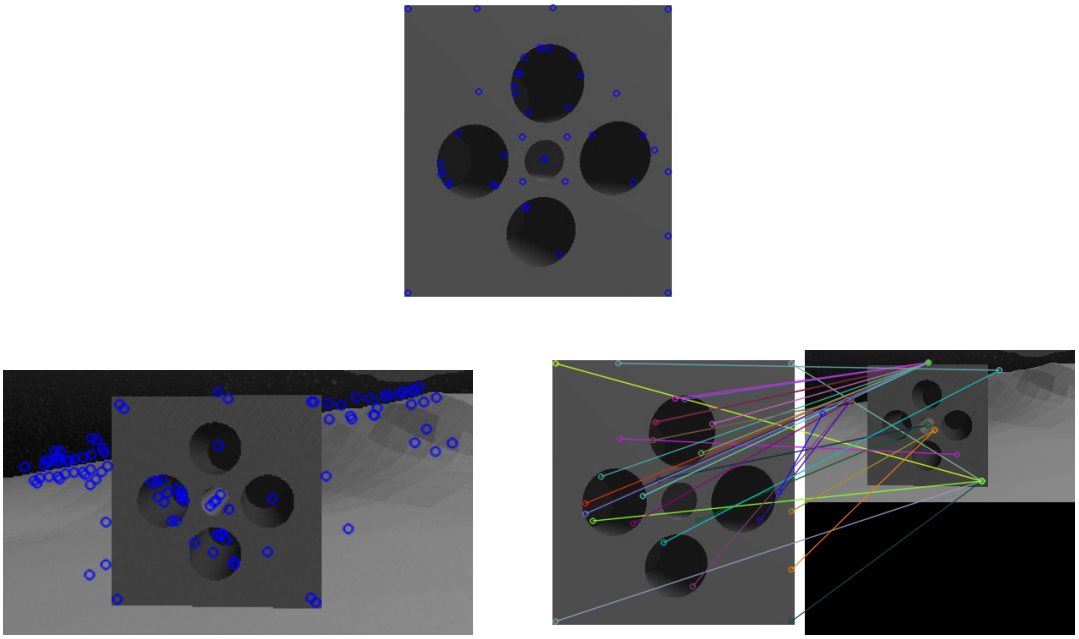

Figure A.5: Result of the algorithm. Above, the detected features (in blue) in the *object image*. Below, on the left, the detected features in the *scene image*; on the right the matched (erroneous) features.

match them, thus, detecting the object in the scene. Then, it is necessary to find the perspective transformation between object image and the scene (i.e. find homography). This is needed to take into account that usually pose and scaling of the object in the scene are not the same of the object image.
The OpenCV tutorial use different tools:

- **SURF** (Speeded Up Robust Features) Detector [Bay *et al.* (2006)] to extract features from *object image* and *scene image*, and to compute descriptors.
- **FLANN** (Fast Library for Approximate Nearest Neighbors) matcher [Muja & Lowe (2012)] to match the features.
- **Lowe's ratio test** [Lowe (2004)]to filter the best matches.
- **RANSAC** (RANdom SAmple Consensus) [Fischler & Bolles (1981)] method to find the homography with the function *findHomography()*.

In this case, results are unsatisfactory as can be seen in A.5. The main problem is that in this particular scene there are not nice distinct features. Also, the symmetry of the structure does not help, because there are a lot of particulars

that are the same (like the square side and the holes). As can be seen in the tutorial, good results are obtained for food boxes. In fact, this methods is often associate to scenes where a lot of details are present (graffiti painting, supermarket shelf, ...). In our underwater case, realistic infrastructures don't have this details.
There are also lot of parameters to set for the three main tools (SURF, FLANN, RANSAC). Various trials have been tried but no-one was satisfactory. Also, different detectors (like SWIFT [Lowe (2004)]) and matchers (like Brute-force), have been tried.
The variety of tools and parameters make this method suitable for a lot of applications, and must be taken into consideration in other applications.

References

Abdullah, M., Roth, H., Weyrich, M. & Wahrburg, J. (2015). An approach for peg-in-hole assembling using intuitive search algorithm based on human behavior and carried by sensors guided industrial robot. *IFAC-PapersOnLine*, **48**, 1476–1481. 7

Antonelli, G. (2009). Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems. *IEEE Transactions on Robotics*, **25**, 985–994. 5

Antonelli, G. & Chiaverini, S. (1998). Task-priority redundancy resolution for underwater vehicle-manipulator systems. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, vol. 1, 768–773 vol.1. 5

Antonelli, G. & Chiaverini, S. (2003). Fuzzy redundancy resolution and motion coordination for underwater vehicle-manipulator systems. *IEEE Transactions on Fuzzy Systems*, **11**, 109–120. 5

Antonelli, G., Arrichiello, F. & Chiaverini, S. (2008). The null-space-based behavioral control for autonomous robotic systems. *Intelligent Service Robotics*, **1**, 27–39. 5

Baerlocher, P. & Boulic, R. (2004). An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. *The Visual Computer*, **20**, 402–417. 6

Bay, H., Tuytelaars, T. & Van Gool, L. (2006). Surf: Speeded up robust features. In A. Leonardis, H. Bischof & A. Pinz, eds., *Computer Vision – ECCV 2006*, 404–417, Springer Berlin Heidelberg, Berlin, Heidelberg. 18

Bingham, B., Foley, B., Singh, H., Camilli, R., Delaporta, K., Eustice, R., Mallios, A., Mindell, D., Roman, C. & Sakellariou, D.

(2010). Robotic tools for deep water archaeology: Surveying an ancient shipwreck with an autonomous underwater vehicle. *Journal of Field Robotics*, **27**, 702–717. 2

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-8**, 679–698. 16

Capocci, R., Dooly, G., Omerdic, E., Coleman, J., Newe, T. & Toal, D. (2017). Inspection-class remotely operated vehiclesa review. *Journal of Marine Science and Engineering*, **5**, 13. 2

Carrera, A., Palomeras, N., Hurtos, N., Kormushev, P. & Carreras, M. (2014). Learning by demonstration applied to underwater intervention. vol. 269. 3

Casalino, G., Angeletti, D., Bozzo, T. & Marani, G. (2001). Dexterous underwater object manipulation via multi-robot cooperating systems. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 4, 3220–3225 vol.4. 3

Casalino, G., Angeletti, D., Cannata, G. & Marani, G. (2002). The functional and algorithmic design of amadeus multirobot workcell. In S.K. Choi & J. Yuh, eds., *Underwater Vehicle Technology*, vol. 12. 3

Casalino, G., Turetta, A., Sorbara, A. & Simetti, E. (2009). Self-organizing control of reconfigurable manipulators: A distributed dynamic programming based approach. In *2009 ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots*, 632–640. 5

Casalino, G., Zereik, E., Simetti, E., Torelli, S., Sperindé, A. & Turetta, A. (2012). Agility for underwater floating manipulation task and subsystem priority based control strategy. In *International Conference on Intelligent Robots and Systems (IROS 2012)*, 1772–1779. 3

Casalino, G., Caccia, M., Caiti, A., Antonelli, G., Indiveri, G., Melchiorri, C. & Caselli, S. (2014). Maris: A national project on marine robotics for interventions. In *22nd Mediterranean Conference on Control and Automation*, 864–869. 1, 4, 7

Centelles, D., Moscoso, E., Vallicrosa, G., Palomeras, N., Sales, J., Mart, J.V., Marn, R., Ridao, P. & Sanz, P.J. (2017). Wireless hrov control with compressed visual feedback over an acoustic link. In *OCEANS 2017 - Aberdeen*, 1–7. 1

Chang, R.J., Y. Lin, C. & S. Lin, P. (2011). Visual-based automation of peg-in-hole microassembly process. *Journal of Manufacturing Science and Engineering*, **133**, 041015. 7

Cheng Chang, C., Yuan Chang, C. & Ting Cheng, Y. (2004). Distance measurement technology development at remotely teleoperated robotic manipulator system for underwater constructions. 333 – 338. 2

Chhatpar, S.R. & Branicky, M.S. (2001). Search strategies for peg-in-hole assemblies with position uncertainty. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 3, 1465–1470 vol.3. 7

Chiaverini, S. (1997). Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, **13**, 398–410. 5

Christ, R. & Wernli, R. (2013). *The ROV Manual: A User Guide for Remotely Operated Vehicles*. Elsevier, 2nd edn. 2

Cieslak, P., Ridao, P. & Giergiel, M. (2015). Autonomous underwater panel operation by girona500 uvms: A practical approach to autonomous underwater manipulation. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 529–536. 4

Di Lillo, P.A., Simetti, E., De Palma, D., Cataldi, E., Indiveri, G., Antonelli, G. & Casalino, G. (2016). Advanced rov autonomy for efficient remote control in the dexrov project. *Marine Technology Society Journal*, **50**. 4

Diaz Ledezma, F., Amer, A., Abdellatif, F., Outa, A., Trigui, H., Patel, S. & Binyahib, R. (2015). A market survey of offshore underwater robotic inspection technologies for the oil and gas industry. 2

Dietrich, F., Buchholz, D., Wobbe, F., Sowinski, F., Raatz, A., Schumacher, W. & Wahl, F.M. (2010). On contact models for assembly tasks: Experimental investigation beyond the peg-in-hole problem on the example of force-torque maps. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2313–2318. 7

Djapic, V., Na, ., Ferri, G., Omerdic, E., Dooly, G., Toal, D. & Vuki, Z. (2013). Novel method for underwater navigation aiding using a companion underwater robot as a guiding platforms. In *2013 MTS/IEEE OCEANS - Bergen*, 1–10. 2

Drap, P. (2012). Underwater photogrammetry for archaeology. *Special Applications of Photogrammetry*. 2

Duda, R.O. & Hart, P.E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, **15**, 11–15. 16

Escande, A., Mansard, N. & Wieber, P.B. (2014). Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *I. J. Robotics Res.*, **33**, 1006–1028. 6

Evans, J., Redmond, P., Plakas, C., Hamilton, K. & Lane, D. (2003). Autonomous docking for intervention-auvs using sonar and video-based real-time 3d pose estimation. In *Oceans 2003. Celebrating the Past ... Teaming Toward the Future (IEEE Cat. No.03CH37492)*, vol. 4, 2201–2210 Vol.4. 3

Evans, J.C., Keller, K.M., Smith, J.S., Marty, P. & Rigaud, O.V. (2001). Docking techniques and evaluation trials of the swimmer auv: an autonomous deployment auv for work-class rovs. In *MTS/IEEE Oceans 2001. An Ocean Odyssey. Conference Proceedings (IEEE Cat. No.01CH37295)*, vol. 1, 520–528 vol.1. 3

Faverjon, B. & Tournassoud, P. (1987). A local based approach for path planning of manipulators with a high number of degrees of freedom. In *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4, 1152–1159. 6

Fischler, M.A. & Bolles, R.C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, **24**, 381–395. 18

Flacco, F., De Luca, A. & Khatib, O. (2012). Prioritized multi-task motion control of redundant robots under hard joint constraints. 3970–3977. 5

Fletcher, B. (2000). Worldwide undersea mcm vehicle technologies. 10. 2

Gilmour, B., Niccum, G. & O'Donnell, T. (2012). Field resident auv systems chevron's long-term goal for auv development. In *2012 IEEE/OES Autonomous Underwater Vehicles (AUV)*, 1–5. 2

Kanoun, O., Lamiraux, F. & Wieber, P. (2011). Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task. *IEEE Transactions on Robotics*, **27**, 785–792. 6

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, **5**, 90–98. 5

Khatib, O. (1987). A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, **3**, 43–53. 4

Lane, D.M., Davies, J.B.C., Casalino, G., Bartolini, G., Cannata, G., Veruggio, G., Canals, M., Smith, C., O'Brien, D.J., Pickett, M., Robinson, G., Jones, D., Scott, E., Ferrara, A., Angelleti, D., Coccoli, M., Bono, R., Virgili, P., Pallas, R. & Gracia, E. (1997). Amadeus: advanced manipulation for deep underwater sampling. *IEEE Robotics Automation Magazine*, **4**, 34–45. 3

Lane, D.M., Maurelli, F., Kormushev, P., Carreras, M., Fox, M. & Kyriakopoulos, K. (2012). Persistent autonomy: the challenges of the pandora project. *IFAC Proceedings Volumes*, **45**, 268 – 273, 9th IFAC Conference on Manoeuvring and Control of Marine Craft. 3

Lee, H. & Park, J. (2014). An active sensing strategy for contact location without tactile sensors using robot geometry and kinematics. *Autonomous Robots*, **36**, 109–121. 7

Lee, J., Mansard, N. & Park, J. (2012). Intermediate desired value approach for task transition of robots in kinematic control. *IEEE Transactions on Robotics*, **28**, 1260–1277. 6

Lowe, D.G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, **60**, 91–110. 18, 19

Lozano-Prez, T., Mason, M.T. & Taylor, R.H. (1984). Automatic synthesis of fine-motion strategies for robots. *The International Journal of Robotics Research*, **3**, 3–24. 8

Maciejewski, A.A. & Klein, C.A. (1985). Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The International Journal of Robotics Research*, **4**, 109–117. 5

Mansard, N., Khatib, O. & Kheddar, O. (2009a). A unified approach to integrate unilateral constraints in the stack of tasks. *IEEE Transactions on Robotics*, **25**, 670–685. 6

Mansard, N., Remazeilles, A. & Chaumette, F. (2009b). Continuity of varying-feature-set control laws. *IEEE Transactions on Automatic Control*, **54**, 2493–2505. 6

Marani, G., Kim, J., Yuh, J. & Chung, W. (2003). Algorithmic singularities avoidance in task-priority based controller for redundant manipulators. vol. 4, 3570 – 3574 vol.3. 5

Marani, G., Choi, S.K. & Yuh, J. (2009). Underwater autonomous manipulation for intervention missions auvs. *Ocean Engineering*, **36**, 15 – 23, autonomous Underwater Vehicles. 3

Marty, P. (2004). Alive: An autonomous light intervention vehicle. *Scandinavian Oil-Gas Magazine*, **32**. 3

Matas, J., Galambos, C. & Kittler, J. (2000). Robust detection of lines using the progressive probabilistic hough transform. *Computer Vision and Image Understanding*, **78**, 119–137. 16

Miura, J. & Ikeuchi, K. (1998). Task-oriented generation of visual sensing strategies in assembly tasks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **20**, 126 – 138. 7

Muja, M. & Lowe, D.G. (2012). Fast matching of binary features. In *Computer and Robot Vision (CRV)*, 404–410. 18

Nakamura, Y. (1990). *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edn. 4

Nakamura, Y. & Hanafusa, H. (1986). Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of Dynamic Systems, Measurement, and Control*. 4, 5

Nenchev, D.N. & Sotirov, Z.M. (1994). Dynamic task-priority allocation for kinematically redundant robotic mechanisms. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*, vol. 1, 518–524 vol.1. 6

Newman, W., Zhao, Y. & Pao, Y.H. (2001). Interpretation of force and moment signals for compliant peg-in-hole assembly. vol. 1, 571 – 576 vol.1. 7

Oh, J. & Oh, J.H. (2015). A modified perturbation/correlation method for force-guided assembly. *Journal of Mechanical Science and Technology*, **29**, 5437–5446. 7

Padir, T. (2005). Kinematic redundancy resolution for two cooperating underwater vehicles with on-board manipulators. vol. 4, 3137 – 3142 Vol. 4. 5

Park, H., Bae, J.H., Park, J.H., Baeg, M.H. & Park, J. (2013). Intuitive peg-in-hole assembly strategy with a compliant manipulator. In *IEEE ISR 2013*, 1–5. 7

Park, H., Park, J., Lee, D.H., Park, J.H., Baeg, M.H. & Bae, J.H. (2017). Compliance-based robotic peg-in-hole assembly strategy without force feedback. *IEEE Transactions on Industrial Electronics*, **PP**, 1–1. 8

Pauli, J., Schmidt, A. & Sommer, G. (2001). Vision-based integrated system for object inspection and handling. *Robotics and Autonomous Systems*, **37**, 297 – 309. 7

Prats, M., Romags, D., Palomeras, N., Garca Snchez, J.C., Nannen, V., Wirth, S., Fernandez, J., P. Beltrn, J., Campos, R., Ridao, P., Sanz, P., Oliver, G., Carreras, M., Gracias, N., Marn Prades, R. & Ortiz, A. (2012). Reconfigurable auv for intervention missions: A case study on underwater object recovery. *Intelligent Service Robotics*, **5**, 19–31. 3

PROMETEO (2016). http://www.irs.uji.es/prometeo/, [online; accessed 25-october-2018]. 4

Ribas, D., Ridao, P., Turetta, A., Melchiorri, C., Palli, G., Fernndez, J.J. & Sanz, P.J. (2015). I-auv mechatronics integration for the trident fp7 project. *IEEE/ASME Transactions on Mechatronics*, **20**, 2583–2592. 1

Rigaud, V., Coste-Maniere, E., Aldon, M.J., Probert, P., Perrier, M., Rives, P., Simon, D., Lang, D., Kiener, J., Casal, A., Amar, J., Dauchez, P. & Chantler, M. (1998). Union: underwater intelligent operation and navigation. *IEEE Robotics Automation Magazine*, **5**, 25–35. 3

ROBUST (2016). http://eu-robust.eu, [online; accessed 25-october- 2018]. 4

Sanz, P., Ridao, P., Oliver, G., Casalino, G., Insaurralde, C., Silvestre, C., Melchiorri, C. & Turetta, A. (2012). Trident: Recent improvements about autonomous underwater intervention missions. vol. 3, 1–10. 3, 6

Schempf, H. & Yoerger, D.R. (1992). Coordinated vehicle/manipulation design and control issues for underwater telemanipulation. *IFAC Proceedings Volumes*, **25**, 259 – 267, iFAC Workshop on Artificial Intelligence Control and Advanced Technology in Marine Automation (CAMS '92), Genova, Italy, April 8-10. 3

Sentis, L. & Khatib, O. (2005). Control of free-floating humanoid robots through task prioritization. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 1718–1723. 5

Shi, J. & Tomasi, C. (2000). Good features to track. *Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **600**, 593600. 15

Shirinzadeh, B., Zhong, Y., Tilakaratna, P.D.W., Tian, Y. & Dalvand, M.M. (2011). A hybrid contact state analysis methodology for robotic-based adjustment of cylindrical pair. *The International Journal of Advanced Manufacturing Technology*, **52**, 329–342. 7

Siciliano, B. & Slotine, J..E. (1991). A general framework for managing multiple tasks in highly redundant robotic systems. In *Fifth International Conference on Advanced Robotics 'Robots in Unstructured Environments*, 1211–1216 vol.2. 5

Simetti, E. & Casalino, G. (2016). A novel practical technique to integrate inequality control objectives and task transitions in priority based control. *Journal of Intelligent & Robotic Systems*, **84**. 4, 7

Simetti, E. & Casalino, G. (2017). Manipulation and transportation with cooperative underwater vehicle manipulator systems. *IEEE Journal of Oceanic Engineering*, **42**, 782–799. 4

Simetti, E., Turetta, A. & Casalino, G. (2009). *Distributed Control and Coordination of Cooperative Mobile Manipulator Systems*, 315–324. Springer Berlin Heidelberg, Berlin, Heidelberg. 5

Simetti, E., Casalino, G., Torelli, S., Sperindé, A. & Turetta, A. (2014a). Floating underwater manipulation: Developed control methodology and experimental validation within the trident project. *Journal of Field Robotics*, **31(3)**, 364–385. 3, 6

Simetti, E., Casalino, G., Torelli, S., Sperindé, A. & Turetta, A. (2014b). Underwater floating manipulation for robotic interventions. *IFAC Proceedings Volumes*, **47**, 3358 – 3363, 19th IFAC World Congress. 7

Simetti, E., Casalino, G., Wanderlingh, F. & Aicardi, M. (2018). Task priority control of underwater intervention systems: Theory and applications. *Ocean Engineering*, **164**, 40 – 54. 4

Song, H., Kim, Y. & Song, J.B. (2016). Guidance algorithm for complex-shape peg-in-hole strategy based on geometrical information and force control. *Advanced Robotics*, 1–12. 7

Sugiura, H., Gienger, M., Janssen, H. & Goerick, C. (2007). Real-time collision avoidance with whole body motion control for humanoid robots. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2053–2058. 5

Suzuki, S. & Be, K.A. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, **30**, 32 – 46. 12, 17

TWINBOT (2019). Twinbot website. 1, 9

Urabe, T., Ura, T., Tsujimoto, T. & Hotta, H. (2015). Next-generation technology for ocean resources exploration (zipangu-in-the-ocean) project in japan. 1–5. 2

Wynn, R., Huvenne, V., Le Bas, T., Murton, B., Connelly, D., Bett, B., Ruhl, H., Morris, K., Peakall, J., Parsons, D., J. Sumner, E., E. Darby, S., Dorrell, R. & Hunt, J. (2014). Autonomous underwater vehicles (auvs): Their past, present and future contributions to the advancement of marine geoscience. *Marine Geology*, **352**. 2

Xu, Q. (2015). Robust impedance control of a compliant microgripper for high-speed position/force regulation. *IEEE Transactions on Industrial Electronics*, **62**, 1201–1209. 8

Yoshikawa, T. (1984). Analysis and Control of Robot Manipulators with Redundancy. In M. Brady & R. Paul, eds., *Robotics Research The First International Symposium*, 735–747, MIT Press. 5

Yuh, J., Choi, S.K., Ikehara, C., Kim, G.H., McMurty, G., Ghasemi-Nejhad, M., Sarkar, N. & Sugihara, K. (1998). Design of a semi-autonomous underwater vehicle for intervention missions (sauvim). In *Proceedings of 1998 International Symposium on Underwater Technology*, 63–68. 3

Zereik, E., Sorbara, A., Merlo, A., Simetti, E., Casalino, G. & Didot, F. (2011). Space robotics supporting exploration missions: vision, force control and coordination strategy for crew assistants. *Intelligent Service Robotics*, **4**, 39–60. 5