Skill Robot Library: Intelligent Path Planning Framework for Object Manipulation

Maria Kyrarini, Sameer Naeem, Xingchen Wang and Axel Gräser
Institute of Automation (IAT), University of Bremen
Bremen, Germany
Email:mkyrar@iat.uni-bremen.de

Abstract— **Commonly used path planning techniques for object manipulation are computationally expensive and time-consuming. In this paper, a novel framework called Skill Robot Library (SRL), which has competence to store only the key-points of a path rather than complete, is presented. The path can be computed with path planner or taught by a human using kinesthetic teaching. Additionally, when the environment is static and only the requested new start and goal positions are changed with respect to the start and goal positions of the stored path, the SRL can retrieve and modify the stored path. The SRL forwards the final path to the robot for reproduction. Experimental results achieved with a six degrees of freedom robotic arm are presented together with performance evaluation of the SRL and the path planner is demonstrated via a series of experiments.**

Keywords—object manipulation; intelligent path planning; robot skill framework

I. INTRODUCTION

Robots are increasingly introduced to cooperate with humans in different fields, such as manufacturing [1] [2], surgical robotics [3], service robotics [4] [5] and search & rescue robotics [6] [7]. One of the main objectives of using robot manipulators in human-robot cooperation is to reliably and repeatedly manipulate objects from a start position to a goal position. Path planning techniques are very often used to achieve the object manipulation, and the main goal is to find a collision free path for a specific environment. This goal is computationally complex due to large amount of data to be processed, which contain the information of the surrounding environment conceived via sensors for processing in order to plan a collision free path [8]. One framework which is widely used in research [10][11] is the Open Motion Planning Library (OMPL) [9] and provides sampling-based path planners, such as Rapidly-expanding Random Trees (RRT)[12], Probabilistic Roadmap Method (PRM) [13] and Kinodynamic Planning by Interior-Exterior Cell Exploration (KPIECE) [14]. The drawback of OMPL Library is that it computes the whole path, even if the environment is static and the start and goal position are slightly changed with a previous computed path.

A framework which learns from experience is implanted by [15] and a complete new path is calculated using OMPL and added to the database. In parallel of calculation of the path, the library manager searches through the library to find a similar path with the least violation of constrains, retrieve it and repair it, based on the new query. Due to this parallel modularity, the system needs high computational power.

The main idea presented in this paper is to develop a low computationally demanding framework, similar to the framework presented in [15], which deals with the time consuming problem of reusing path planner and can be used by robot manipulators with different number of degrees of freedom. The proposed solution is an intelligent algorithm called Skill Robot Library (SRL) that has the capability to store new paths, computed either from OMPL or demonstrated via kinesthetic teaching [16] in a library (database) by recording the signals of the joint Configuration (C-space) and the calculated by forward kinematics Cartesian path, and to reuse an already calculated stored path on user request. Additionally, when requested new start or/and goal positions are in an area close to the stored start and goal position and the environment is static, the proposed framework SRL is able to adapt the path to the request, without re-computing the whole path.

However, searching a path from a database with large amount of data is also cumbersome and computationally expensive. There is a need to limit the number of points of each trajectory, similar to the work presented in [17], so that only the key-points, which are most important for reproduction of the trajectory, are stored in the library. A novel application of the polyline simplification algorithm by Lang [18] is introduced in the presented work, which finds the key-points on-line of each new path and stores only those key-points in the library. Important key-points are set of points that are the most important for reproduction of the path by the robot manipulator. The simplification algorithm uses as input the path in Cartesian space. The output of the simplification algorithm is the key-points of the path and together with the corresponding joint configuration of the key-points are stored in the SRL.

The paper is organized as following; in section II and III the Skill Robot Library and its functionality are described, in section IV the experimental results are presented and in section V conclusion and future work are included.

II. SKILL ROBOT LIBRARY (SRL)

The block diagram in Fig.1 gives the overview of our proposed framework, the Skill Robot Library (SRL). In this section the modules of the block diagram are explained. The

ISBN 978-0-9928626-7-1 © EURASIP 2017

SRL is implemented using ROS (Robot Operating System) [19].

A. Path Library Manager (PLM)

The Path Library Manager (PLM) is a decision-making module and has dual functionality. The first functionality of this module is to decide whether a stored path should be used or a new path should be computed by the planner. The PLM searches if the start position in C-space (current position since the robot manipulator is already in start position) is within a tolerance distance from a point stored in the database. If PLM finds paths that satisfy those criteria, then, the PLM searches only those paths if the goal position in Cartesian space is within tolerance distance defined by the user, from a point stored in those paths. If a path or a segment of a path is found, the PLM confirms that there is no collision with the environment and forwards it to the module of *retrieve, modify and reconstruct path*. When the PLM has more than one solutions (more than one points that are close to the requested start or goal position, or more than one paths satisfied the above criteria), always it selects the path or segment of path with the smallest difference between the start position and a stored point in C-space and the shortest distance between the goal position and a stored point in Cartesian space. The decision criteria are explained in detail in section III. The search in the path library database is bidirectional, in same or reverse direction. If no stored path is found or there is a collision with the environment from the retrieved path, a new path will be computed by the planner. Obstacle positions in the environment are perceived by a vision sensor and used for collision detection, as explain in the previous work [16].

The second functionality of PLM is to define whether a path should be stored or not in the Path Library Database. Furthermore, PLM manages all the data related issues and all the search operations needed to locate a specific path whenever the user needs to send the manipulator to another location.

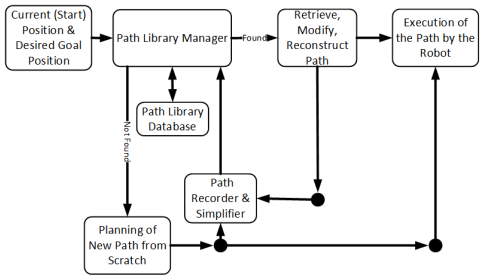

Figure 1: Overview of the Skill Robot Library (SRL)

B. Planning of New Path

When there is a need to move the robot manipulator from one initial position to another goal position the path has to be calculated using one of the planners available in OMPL or using kinesthetic teaching. The OMPL provides the user with different choices of path planners. In this paper, the RRTConnect (Rapidly-exploring Random Trees Connect) [20] path planner is selected.

The example of planning a collision free new path using RRTConnect from start point S to goal point G is shown in Fig.2a. The planned path plotted in Fig.2b shows the curves which are due to obstacle avoidance. The same path will be used throughout the paper to explain different cases. The path consists of numerous waypoints depending on the distance between the initial and the goal position, complexity of the available Inverse kinematic (IK) solutions, and other constraints such as obstacles in the surrounding. The single path shown in Fig.2b can be broken into its individual waypoints (Fig.2c).

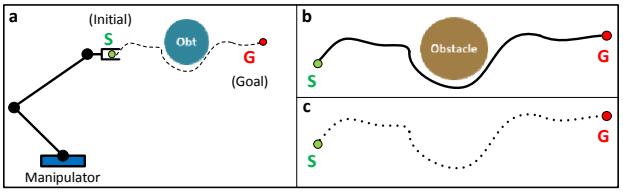

Figure 2: (a) the start point ***S*** and the goal point ***G*** for which the path is planned. (b) the planned path from start point ***S*** to the goal point ***G***. (c) the path fragmented into individual waypoints while preserving the uniqueness.

C. Path Recorder and Simplifier

Once the path has been calculated by the planner, the waypoints are published by the robot's joint states to provide input to the joint controllers to command the robot. The joint states are published at a frequency of 10Hz [21] and then filtered as there is excessive number of waypoints in a single path. Firstly, a filter is applied to eliminate the waypoints which are similar to the previously received ones or have the difference less than 0.001 radian difference (~0.05 degrees). However, after this filtering the number of the waypoints is still large. For the above reason, the polyline simplification algorithm by Lang [18] is used in order to find the key points of the path online. A polyline is a connected sequence of line segments or/and arc segments [22]. The Lang simplification algorithm is used in cartography [23] to simplify polylines. The algorithm is implemented as follows:

- A size-fixed search region is defined and a segment is created between the first and last points of that region. The first point is added in the simplified polyline. The perpendicular distance is calculated between the segment and each intermediate point.
- If any calculated distance is larger than a specified tolerance, the search region in narrowed by excluding the last point of the segment.
- The above process continues until all calculated distances are smaller than the specified tolerance, or there are no more intermediate points. When this condition is fulfilled, all intermediate points are removed, and the first and last points of the segment are part of the simplified polyline.
- A new search region is defined starting at the last point from the previous search region, and the above process is repeated until the whole original polyline is simplified.

D. Path Library Database

In the Path Library Database, the simplified paths are stored after the decision of the PLM. The paths are stored both in Configuration space and in Cartesian space calculated by forward kinematics. Once a path is stored, it remains always there unless amended otherwise by the user.

ISBN 978-0-9928626-7-1 © EURASIP 2017

E. Retrieve, Modify and Reconstruct Path

This is one of another essential part of the SRL in which the simplified stored path has to be reconstructed without using the path planner. This module is responsible for retrieving, modifying (if needed) and reconstructing the path. The section III explains in details how to retrieve and modify paths. Once the stored path is retrieved and modified, the corresponding joint Configuration of the way points of the path is sent to the joint controllers and the path is reproduced by the robot manipulator in desired speed.

III. Functionality of Skill Robot Library (SRL)

In the previous section all the major parts of implementation of the SRL have been presented. In this section a detailed description of the functionality of skill robot library is presented. In fact, a stored single path can be used in different possible scenarios to reconstruct the desired path, but only some representative scenarios are discussed in this paper. The algorithm 1 shows the concept of the SRL for searching, retrieving and modifying a path. The symbols S and G represent start and goal position, respectively, of the original simplified path stored in the library database, and they will be called original start and goal for simplicity. The symbols S' and G' represent the new start and new goal position, respectively, required to move the end effector (EE) of the robot's manipulator. Further, P is the stored path and P' is the new path to be reconstructed using path library. In the algorithm 1 subscript J denotes "with respect to Joint space" and EE denotes "with respect to End-Effector Cartesian pose". P_i is the i-th path, where $i = 1, \dots, n$, and n is the total number of paths stored in the SRL.

A. Search, Retrieve, Reconstruct and Execute Path

This part of the algorithm is responsible for searching the requested path in the library database whenever the manipulator is to be moved from any initial position S' to any goal position G'. If the current joint space configuration of robotic arm is S'_J and $S'_J \in P_i$ for one of the many trajectories in the library then the desired goal position G'_{EE} is checked in that particular path, which if $G'_{EE} \in P_i$ is retrieved and consequently resulted in reconstruction of the requested path since $P' \subseteq P$. This is further divided into two parts, first if P' is improper subset of P ($P' \subseteq P$), as shown in the Fig.3a,b. Secondly, when only the partial path of the stored is used i.e. $P' \subsetneq P$ as shown in the Fig.3c,d,e,f. Moreover the stored path can be used in forward (Fig.3a,c,e) and reverse (Fig.3b,d,f) directions for each of the case discussed in this paper.

The blue colored paths in Fig.3 and Fig.4 show the new path to be planned between new start and goal (i.e. S' & G') and the black paths are the remaining original simplified paths stored in the library database which remains unused in some cases. The arrows represent the direction of path followed.

An auto-correction mechanism is also developed in this algorithm without the use of planners when the S & S' and G & G' lie within a tolerance area for each joint (in the presented approach a tolerance area of 3 degrees is selected). If the joint angle for the stored start position S and the new start position S' is within a specified tolerance (in the presented approach the tolerance is 1°), then there is no need to plan the path using the planner instead adjust the joints using joint controller and retrieve the stored path from the library. Note that this is just an adjustment tolerance and is not in any way connected to what is to be discussed in the next section III-B.

Algorithm 1: Functionality of the SRL

```
1   if S'_J ∈ P_i
2    for all P_i do
3        if G'_EE ∈ P_i ;
4            P' ← P_i(S'_J , G'_J);
5             if P' ≠ ∅ then
6                    return P';
7             end
8        else if G'_EE ∈ R_M
10                   extend & modify P_i | P' ⊆ P_i && G'_J ∈ P_i ;
11                   P_n ← P';
12                   return P';
13                   end
14  else use planner;
```

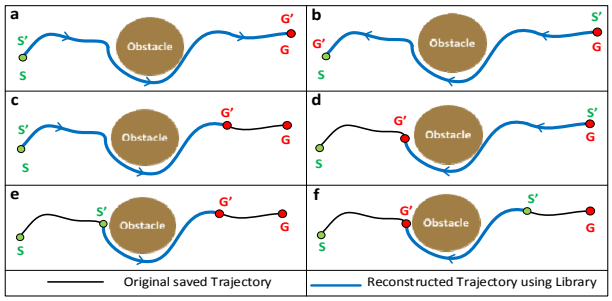

Figure 3: (a-f) Six different possibilities to retrieve the whole stored path or a segment of it. The path reconstructed from $\boldsymbol{S'}$ to $\boldsymbol{G'}$; where $\boldsymbol{S'}$ & $\boldsymbol{G'}$ are the new start and goal positions for the path and $\boldsymbol{S}$ & $\boldsymbol{G}$ are the original start and goal positions.

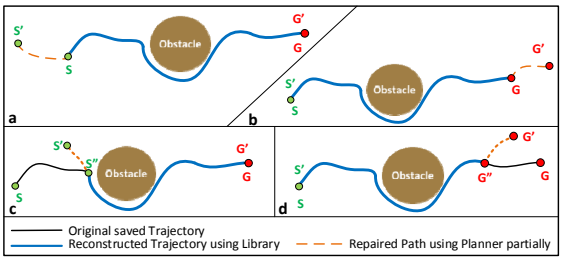

Figure 4: (a) Path is repaired by first using the planner from $\boldsymbol{S'}$ to $\boldsymbol{S}$ and then from $\boldsymbol{S}$ to $\boldsymbol{G'}$ the stored path from library is executed (b) Path from $\boldsymbol{S}$ to $\boldsymbol{G}$ the stored path from library is executed and then the planner is used to compute the path from $\boldsymbol{G}$ to $\boldsymbol{G'}$ (c) A new point $\boldsymbol{S'}$ is found somewhere near the segment of the stored path which is feasible to repair since falling into the modifiable region from $\boldsymbol{S'}$ to $\boldsymbol{S''}$ and then execute the stored path from $\boldsymbol{S''}$ to $\boldsymbol{G'}$/ $\boldsymbol{G}$ (d) Path from $\boldsymbol{S}$/$\boldsymbol{S'}$ to $\boldsymbol{G''}$ the segment of stored path from library is executed and then the planner is used to compute the path from $\boldsymbol{G''}$ to $\boldsymbol{G'}$

B. Search, Retrieve, Modify, Reconstruct and Execute Path

This part of the algorithm runs simultaneously along the parts discussed in section III-A, but is only executed if the conditions of the above described methods are not satisfied. It is responsible for searching the new start S' and/or new goal G' around any point of the paths stored in the library database. Essentially a region is defined here called modifiable bounded region (R_M), which determines whether the stored path can be

ISBN 978-0-9928626-7-1 © EURASIP 2017

modified or not. Supposing that the new path is to be planned from S' to G' which is not completely similar to the stored path P but only a segment of the stored path can be used. This means that, either S' or G' is found in the stored path P i.e. ($S' \in P$ and $G' \notin P$) or ($G' \in P$ and $S' \notin P$). But S' or G' lie within the defined modifiable region for the reconstruction, as shown in Fig.4. Note that before in section III-A the path that needed to be planned was some segment of the stored path but here the stored path is being used as partial part of new path P' and the new path is not exceeded the old path (S to G), which means that S' & G' were the subset of S & G, while here $P' \nsubseteq P$.

Once the S' or G' is detected within defined modifiable region (in the presented approach the modifiable region is 0.30m) then the RRTConnect planner is called to plan the path for either S' to S or G to G' (shown with orange dashed line in Fig.4). After that the stored path of the library is followed (shown in blue color line in Fig. 4). Both parts are integrated to work in collaboration with each other. After the execution the new path is saved in the SRL either as a new or included in the same store path depending on the different scenarios. For example, in scenarios shown in Fig.4a and b the same path is amended as there is no harm to elongate the path from beginning or end with useful waypoints. In Fig.4c and d the path is saved as a new path because it is not preferable to insert anything in between the existing stored path.

IV. Results and Evaluation Performance

Experiments have been performed on Universal Robot UR10 (http://www.universal-robots.com/products/ur10-robot/) manipulator; where the number 10 represents the payload in kilograms that the arm can carry. The UR10 robot manipulator has 6 degrees of freedom and it is driven by gravity-compensation controllers, which enables also kinesthetic teaching, as shown in Fig.5.

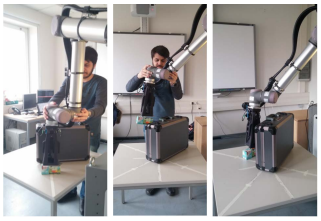

Figure 5: The pick and place scenario with kinesthetic teaching. First the human guides the robot to pick the object (left image) then using kinesthetic teaching (middle image) place the object at the desired position (right image) avoiding the obstacle.

The results shown here represent the path for pick and place scenario, in which the robot is first at the home position that is the up position as seen in Fig.6a and then moves towards the object of interest, grasps it and places it on the desired goal location. For the Lang algorithm, the search region has been selected by the user equal to 15 points and the tolerance 0.1m. The different scenarios as explained in section III are shown with actual data plotted on 3D graphs in the Fig.6a-d. The recorded path, shown in Fig.6a, consists of C-space and Cartesian Path calculated from forward kinematics, but it is difficult to visualize the joint space in 6D and therefore the graphs plotted are based on 3D Cartesian space (x, y, z). The graphs (Fig.6a-d) show the scattered plots instead of continuous since we consider the waypoints for plotting. The paths from stored start point S to stored goal G in dull red dots show the original stored path's waypoints, while the path in blue dots shows the stored path after simplification.

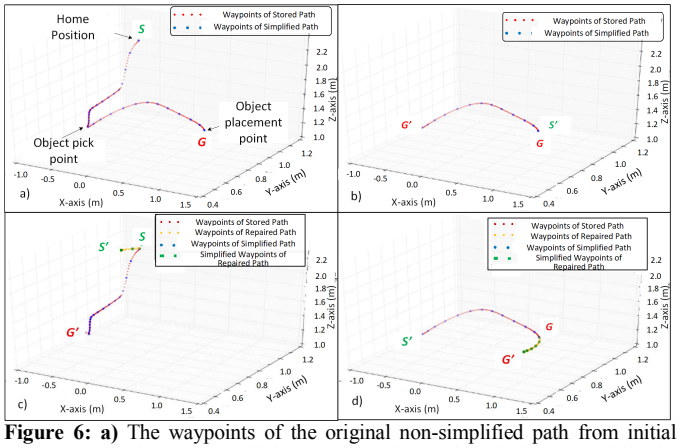

Figure 6: a) The waypoints of the original non-simplified path from initial point S to final target G, which are 281 (red) for this single path, while the simplified path consists 30 waypoints (blue). **b)** The partial path of the stored path is used. In this case the new goal position (G') is intermediate point of the stored path and the new start position is the same as the stored goal position ($S'=G$). Moreover, path is followed in reverse direction. **c)** Modified path from new start position (S') till the stored start position (S) in orange and then the partial stored path from S to intermediate point G' is followed. The green square symbols show the simplified and extended path, while orange ones are the original planned path for modification. **d)** Modified Path from S' to G', first using the stored path partially from intermediate point from store path S' to G and then using the planner from stored goal (G) to new goal position (G').

The graph in Fig.6b shows how the partial path in used from the stored path in the library database. The new path from S' to G' is reconstructed using the simplified waypoints only, but in reverse direction, i.e. the new start S' is equal to stored goal G (i.e. $S' = G$). The new goal G' is the intermediate point found in the stored path. In red are the waypoints from the original path that are 161 waypoints and in blue are 13 waypoints from the simplified path. The graph in the Fig.6c illustrated that the path is modified using planner from S' to S in the beginning and then partial part of the stored path from the path library is followed i.e. from S to G', where G' is new goal point found at some intermediate point of saved path. Here the partial original path after extension consists of 123 waypoints, while simplified consists of 17 waypoints. The graph of Fig.6d illustrates that the path now is modified first by following the stored path partially from some intermediate point S', which has the new start position and then using planner from G to G'. Here the partial original path after extension consists of 173 waypoints, while simplified consists of 19 waypoints.

The application of the developed algorithm is not only limited to intelligent path planning but also contributes to single demonstration kinesthetic teaching (Fig.5). One of the example paths is shown in Fig.7a where the human guides the robot's end-effector through a new movement from point S to G while avoiding the obstacle. In Fig.7b the path, that the robot learned by the human and was stored in the SRL, is modified using the planner from G to G'.

ISBN 978-0-9928626-7-1 © EURASIP 2017

A factual comparison is presented in Fig.8 between the planning & execution time of the path planner against the developed algorithm which reconstructs the path using the path library. The computation power of the system used for the results is a computer with CPU core i3 of processing speed 1.9 GHz and memory 4 GB. As it can clearly be seen that the implemented algorithm is less time-consuming in comparison to path planning and the failure rate against the path planner is zero, as long as the path or segment of it is stored in the library.

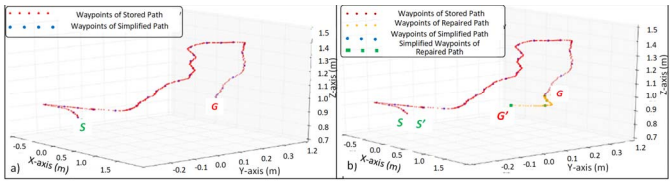

Figure 7: a) Path stored using kinesthetic teaching to teach the real robot to pick and place the object using human demonstration. There are 571 waypoints of the original path, whereas the simplified path consists of 46 waypoints. **b)** Path of kinesthetic teaching (S to G) modified (orange) to reach another nearby goal position G'.

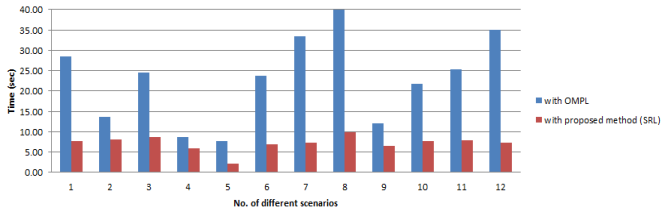

Figure 8: The graph for comparison between the path planned & executed by the planner and the stored path used for reconstructed using Skill robot library for different scenarios.

V. Conclusion

Object manipulation is a challenging task and the path planners are widely used. The proposed framework SRL is able to store the important key-points of the path, retrieve and modify the stored path based on the task requirements. Additionally the SRL can store trajectories created by the path planners and taught trajectories of kinesthetic teaching. Moreover, due to the path simplifier, the computational power needed is low, as it can be seen from the evaluation performance. The SRL framework needs less time for execution, in comparison to the RRTConnect path planner, as shown in Fig.8. Last but not least, the SRL is a generic algorithm and it can be used with various robot manipulations having different degrees of freedom. In future work, the framework will be extended to support dual-arm robots and, additionally, non-stationary environments by modifying the intermediate key-points.

References

[1] C. Schou, J. Damgaard, S. Bogh and O. Madsen, "Human-robot interface for instructing industrial tasks using kinesthetic teaching," *2013 44th Int. Symposium on Robotics (ISR)*, Seoul, 2013.

[2] S. Nikolaidis and J. Shah, "Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy," *2013 8th ACM/IEEE Int. Conf. on Human-Robot Interaction (HRI)*, 2013.

[3] M. D. Comparetti, E. Beretta, M. Kunze, E. D. Momi, J. Raczkowsky and G. Ferrigno, "Event-based device-behavior switching in surgical human-robot interaction," *2014 IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014.

[4] Q. Zhao, D. Tu, S. Xu, H. Shao and Q. Meng, "Natural human-robot interaction for elderly and disabled healthcare application," *2014 IEEE Int. Conf. on Bioinformatics and Biomedicine (BIBM)*, 2014.

[5] A. Graser, T. Heyer, L. Fotoohi, U. Lange, H. Kampe, B. Enjarini, S. Heyer, C. Fragkopoulos and D. Ristic-Durrant, "A Supportive FRIEND at Work: Robotic Workplace Assistance for the Disabled," *IEEE Robotics & Automation Magazine,* vol. 20, no. 4, pp. 148 - 159, 2013.

[6] R. Rocha, D. Portugal, M. Couceiro, F. Araujo, P. Menezes and J. Lobo, "The CHOPIN project: Cooperation between human and rObotic teams in catastrophic incidents," in *2013 IEEE Int. Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2013.

[7] H. Al Tair, T. Taha, M. Al-Qutayri and J. Dias, "Decentralized multi-agent POMDPs framework for humans-robots teamwork coordination in search and rescue," in *2015 Int. Conf. on Information and Communication Technology Research (ICTRC)*, 2015.

[8] K. Krishnaswamy, J. Sleeman and T. Oates, "Real-Time Path Planning for a Robotic Arm," in *PETRA 2011 Proceedings of the 4th Int. Conf. on PErvasive Technologies Related to Assistive Environments*, 2011.

[9] I. A. Sucan, M. Moll and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine,* vol. 19, no. 4, pp. 72 - 82, 2012.

[10] S. Paulin, T. Botterill, J. Lin, X. Chen and R. Green, "A comparison of sampling-based path planners for a grape vine pruning robot arm," in *2015 6th Int.l Conf. on Automation, Robotics and Applications,* 2015

[11] D. Schneider, E. Schomer and N. Wolpert,"Completely Randomized RRT–Connect: A Case Study on 3D Rigid Body Motion Planning," in *2012 IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2015

[12] I. A. Şucan and L. E. Kavraki, "A sampling-based tree planner for systems with complex dynamics," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 116–131, 2012.

[13] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation,* vol. 12, no. 4, pp. 566–580, 1996.

[14] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research (IJRR)*, vol. 20, no. 5, pp. 378–400, May 2001.

[15] D. Berenson, P. Abbeel and K. Goldberg, "A robot path planning framework that learns from experience," in *2012 IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[16] M.Kyrarini, A.Leu, D.Ristić-Durrant, A.Gräser, A.Jackowski, M. Gebhard, J.Nelles, C.Bröhl, C.Brandl, A.Mertens, C.M.Schlick, "Human-Robot Synergy for cooperative robots", 2016, *Facta Universitatis,series:Autonomic Control and Robotics*,15(3),pp. 187-204.

[17] M. Nagahara and C.F. Martin, "L^1 Control Theoretic Smoothing Splines," *IEEE Signal Processing Letters,* 21(11), pp.1394-1397, 2014.

[18] T. Lang, "Rules for robot draughtsmen," *The Geographical Magazine,* 42(1), pp. 50-51, 1969.

[19] M. Quigley, B. Gerkeyy, K. Conleyy, J.Fausty, T. Footey, J. Leibsz, E. Bergery, R. Wheelery, A. Ng, W. Garage and M. Park, "ROS: an open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, 2009.

[20] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2000.

[21] "Joint State Publisher - 3.1.2 Parameters," ROS, [Online]. Available: http://wiki.ros.org/joint_state_publisher. [Accessed 29.01.2017].

[22] J. Xiahou, J. Rao and M. Tong, "A New Approach of Parametric Auto-generation on Polyline," in *2010 2nd Int. Conf. on Software Technology and Engineering (ICSTE) (Volume:1)*, 2010.

[23] R. Mc Master, "The Integration Of Simplification And Smoothing Algorithms In Line Generalization," *Cartographica: The Int. J. for Geographic Information and Geovisualization,*26(1), pp. 101-121, 1989.