Humanoid Robot Path Planning Using Rapidly Explored Random Tree and Motion Primitives

Submitted: 6th April 2020; accepted: 14th March 2021

Maksymilian Szumowski, Teresa Zielinska

DOI: 10.14313/JAMRIS/1-2021/3

Abstract:
Path planning is an essential function of the control system of any mobile robot. In this article the path planner for a humanoid robot is presented. The short description of an universal control framework and the Motion Generation System is also presented. Described path planner utilizes a limited number of motions called the Motion Primitives. They are generated by Motion Generation System. Four different algorithms, namely the: Informed RRT, Informed RRT with random bias, and RRT with A like heuristics were tested. For the last one the version with biased random function was also considered. All mentioned algorithms were evaluated considering three different scenarios. Obtained results are described and discussed.*

Keywords: *humanoid robot, path planning, rapidly exploring random tree*

1. Introduction

An universal control framework for a humanoid robots performing various task is proposed. Despite of many solutions designed for specific robots, universal and simple but effective framework is still an actual research topic. Such need was demonstrated by DARPA Robotics Challenge [6] where humanoid robots were acting semi–autonomously in disaster scenario, or by Robo–Cup Soccer Humanoid League [10] where humanoids were competing with each other.

Essential part of a control system of any mobile robot is a path planner. Presented work describes developed and tested Motion Generation System (MGS) for a humanoid robots. Joint trajectories generated by the Motion Generation System that describe one fragment of the robot's motion are called the Motion Primitive. Motion Primitive transfers the robot from one localisation to another while maintaining the postural balance and fulfilling the joints constrains for position, velocity and torque. MGS can either be implemented as on–line generator, or can be used off–line pre–generating the gait cycles. Off–line mode reduces the needed for computational power of a robot controller. However not every motion can be pre–determined, maneuvering in dynamic environment cannot be left as an off–line option. Path planning can be viewed as a search in a configuration space. For this purpose the search trees are constructed reflecting transformations from the initial to goal configurations. The Rapidly–exploring Random Trees (RRT) is very popular motion planning method applied in mobile robots. In this method first the graph of possible paths is created and next it is determined a feasible but not necessarily optimal path between the initial and final localisation. RRT* is the modified RRT algorithm that aims to obtain the shortest path in a term of some defined metric. Informed RRT* is the modification of RRT* which behaves as RRT* until a first solution is found. Afterwards is only sampled the subset of configurations defined by an admissible heuristic for possible solution improvement. It uses the heuristics for shrinking the planning problem only to the subsets of the original domain [4]. Due to limited number of possible movements, the Informed RRT and not Informed RRT* is used. The difference lies in post processing which can be applied to smooth out generated tree.

In this work the RRT approach is applied for path planning considering the obstacles. The Motion Primitives are used. Presented case assumes that the robot walks on the 2D plane perpendicular to the gravity vector. Two different methods were analysed: classic RRT with informed tree and RRT with A*–like heuristics. Both methods are additionally studied with biased towards the goal configuration the function is used to build the tree.

The paper is structured as follows. After discussing the related works, the path planner description taking into account the robot tasks (section 3) is given. Section 4 describes details of path planning method. In section 5 experimental results demonstrating advantages of presented approach are presented. At the end the conclusions are given.

2. Related Work

Path planner usually makes a separate module (or layer) in a structured (usually hierarchical) control architectures. In article [11] authors presented the hierarchical control system for planning and simulating dynamic motions of a humanoid robots. The sampling–based motion planning method is here applied. Method concerns the motion in limited configuration space. The effectiveness of Bi–directional Rapidly–explored Random Tree approach was proven even for imposed kinematic/kinodynamic constrains. By contrast in presented publication the path planning considering different configuration space and localisation of the whole robot in the global reference frame is studied.

Motion planning for humanoid robots (or any walking robots) is very actual research topic [9], [1]. In article [1] researchers provide analysis of use different tree building algorithms: RRT–CONNECT and RRT–

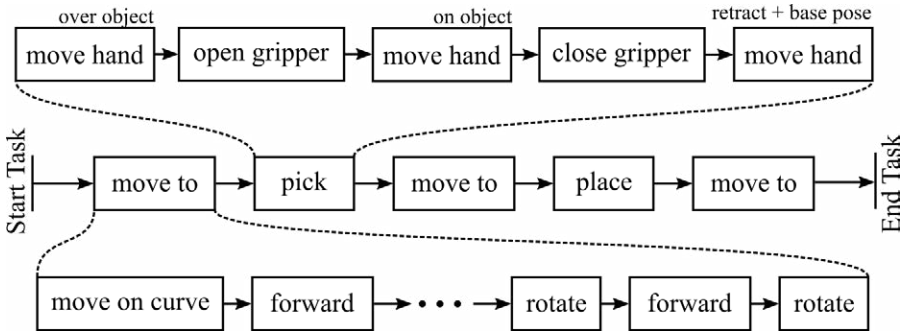

Fig. 1. Task division into motion primitives

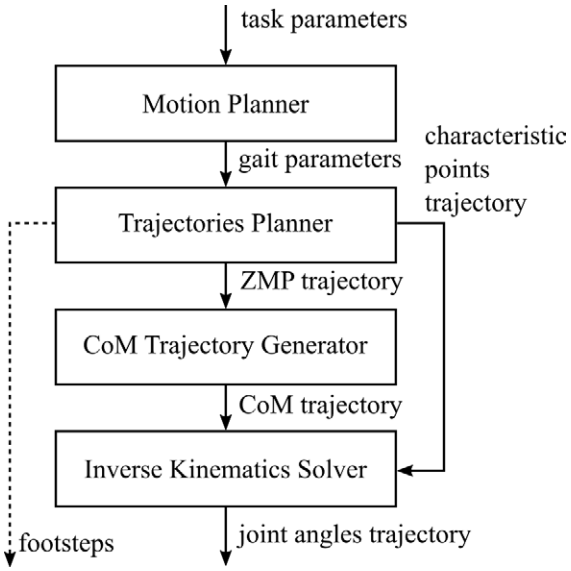

Fig. 2. The structure of Motion Generation System

EXTEND. RRT–CONNECT tries to connect two trees, one constructed starting from the initial configuration and another one built starting from the goal. In each the so called sample the modified configuration is checked according to the tree structure. Difference between RRT–CONNECT and RRT–EXTEND lies in restriction on number of samples (RRT–CONNECT has no restriction). Proposed algorithms (both cases) fulfilled the expectations but there is still room for improvement. Presented algorithms are not performing well when initial and final configuration are close but the obstacles exist. In this work we analysed 3 different cases (goal configurations) which are of the above type. Additionally, not only classic Euclidean norm as a distance measure (distance between the points in 2–D plane) is analysed but the converge rate (time of path evaluation) for finding best possible motion is studied.

The Rapidly-exploring Random Trees (RRT) [7] is a very popular motion planning method applied in mobile robots. Often the methods used for mobile (wheeled) robots path planning are transferred to the legged robots. The example are the methods presented in [12] and [2], [17]. In [12] the RRT method was applied for 6-legged robot. Described in these publications algorithms are providing interesting improvements however they have the drawbacks as well. In [12] the user must specify additional inter–mediate configurations what is limiting the automatic path planning. As studied in [2] an on-line motion generator is still needed to generate additional motions that were not previously specified. However, the algorithm is quite universal and allows to design collision–free, kinematically–feasible paths for any shape of a vehicle. In presented work the off–line pre–generated Motion Primitives are used and no additional motion generation procedures are needed.

The RRT* algorithm was used for path planning of a mobile robots [15], [3] and providing a complete and optimal solution. Unfortunately RRT* is not useful when the tasks or space is limited (limited motion set). Therefore the RRT* algorithm is not applicable in our research but RRT–A* is. In article [8] authors introduced A–Star(A*) cost function to RRT algorithm. Similar approach is used in presented method with adapting it to the known motions set.

The modified RRT method is applied for path planning of a humanoid robot in cluttered environment. The existing state–of–the–art approach is improved using the limited set of motion primitives. After the valid paths are found additional conditions are applied for excluding from the tree search the nodes without the improvement of assumed measure/criterion.

3. General Concept of Path Planning

When designing the path planner the a broader context must be taken into account. — The user defines the task for a humanoid robot. For example the task can be "pick an object from the table and place it on the shelf". Figure 1 presents scenario for pick–and–place task divided into motion primitives. In this case user defines the task as a set of smaller sub–tasks: "move robot to certain location", "pick an object", "move robot to another location", "place an object" and "move robot to final location". For simplicity, Fig.1 is not including the error handling and the optional cases, eg. "object is not available to pick". The sub–tasks are further divided into motion primitives. While "pick" and "place" sub–tasks can be defined using pre–defined template, "move to" sub–tasks significantly differ from each other thus cannot be defined from template. Moving the robot from one place to another requires the work of the path planner. The path planner should take into account not only obstacles avoidance but also kinematic constraints of the robot. It

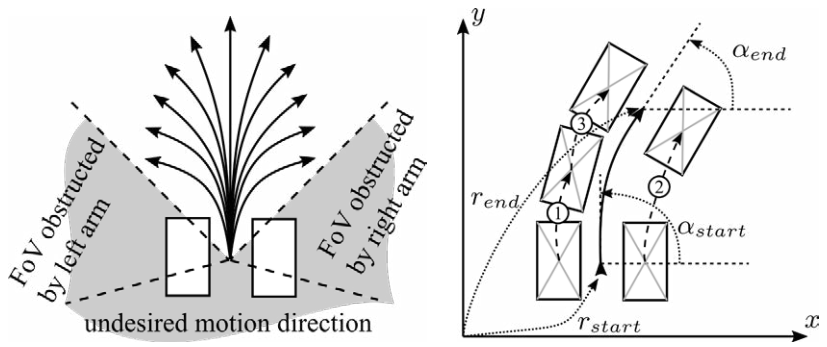

Fig. 3. Illustration of applied Motion Primitives (left picture) and example of foot placement for selected Motion Primitive (right picture)

Tab. 1. Motion Primitives used in path planning algorithm. $r_{start} = [0, 0']$ and $\alpha_{start} = 0°$

	$r_{end}[mm]$	$\alpha_{end}[°]$	time [s]	steps	$V_{mean}[\frac{mm}{s}]$
1	$[60, 0]$	0	11.5	2	5.2174
2	$[120, 0]'$	0	16.5	3	7.2727
3	$[40, 40]$	45	36.5	7	1.5498
4	$[40, -40]$	-45	36.5	7	1.5498
5	$[50, -30]$	30	26.5	5	2.2004
6	$[50, 30]$	-30	26.5	5	2.2004

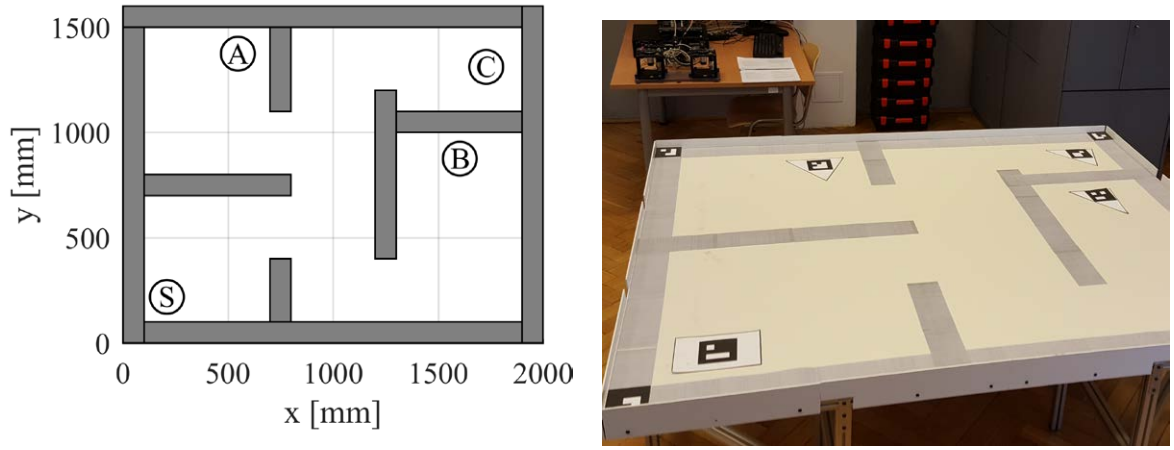

Fig. 4. Generated map with (left picture) from real scenario (right picture)

should be also possible to modify a path in the real-time when an unexpected obstacle appears.

Utilization of motion primitives in path planning drastically reduces the size of required on-board memory and computational demand. Let us consider the area of $1.6m \times 2m$ as our robot environment. The area is surrounded by walls and the robot is $55cm$ tall. Covering the whole environment by RRT is needed approximately 2000 edges. Each edge takes from $8s$ up to $20s$ (approximately $14s$) to be performed. The sampling rate of generated motion by MGS is $1kHz$. The robot uses 20 actuators, the control signal for each motor has resolution of 16 bits. Without optimization and not utilizing motion primitives it is needed $2000 \times 14 \times 1000 \times 20 \times 16 = 8960000000bits \approx 1.12TB$ of stored data. Not only storing but operating such amount of data would be the considerable task. Of course such method of data representation, is not used in real–life applications. Such simple example illustrates the disadvantage of poorly designed methods and emphasise the need for careful selection of motion primitives and demand for the efficient motion planning algorithms.

In this work six motion primitives are applied. Just for those motion primitives $6 \times 14 \times 1000 \times 20 \times 16 = 3.36MB$ data are needed. If the motion primitives are generated off–line they can be stored and then used as a read–only data. In the path planning, as the Motion Primitives the initial and final pose are used. These poses must be defined. Moreover the added around space regions used in collision detection function (a part of path planning algorithm) must be given. Such definitions improve efficiency of path planner.

When generating motion primitives the previously designed and evaluated Motion Generation System (MGS) [14] and [13] is used. Structure of this system is presented in figure 2. Motion Primitives are using the *motion parameters*. After planning the feet placements by Motion Generation System (MGS), the feet transfer trajectories, trajectories of hands and ZMP reference trajectory are produced. Basis on reference trajectory of ZMP (Zero Moment Point) the CoM (Center of Mass) trajectory is obtained for which the ZMP criterion [16] is fulfilled. An inverse kinematic problem is solved and angular trajectories of the joints are obtained. Foot placements (footsteps) are produced by path planning algorithm in Trajectories Planner module.

Applied Motion Primitives are presented in Fig. 3. Left part of figure presents the available motions. They are defined taking into account the unidirectional on–board vision system. Such movements are possible that the explored field of view (FoV) allows to detect the obstacles. In the right side of the figure the Motion Primitives as the sequence of steps are shown. The initial (r_{start},α_{start}) and final position and orientation is denoted (r_{end},α_{end}). For each Motion Primitive initial and final pose of the robot is the same – it is symmetrical towards the saggital plane of the robot with holding the feet parallel to each other. This allows immediately start the next Motion Primitive after the previous ends.

Mean travelling distance for all Motion Primitives equals $4.13cm$. Radius of acceptable tolerance towards the goal position is equal to $10cm$ (it is a bit more than 2 times bigger than the mean distance travelled).

It must be noted that selecting too small number of Motion Primitives can cause various problems. First of all, due to the limited selection of available motions the overall robot path may become longer in terms of a distance and time than the path designed using more Motion Primitives. Secondly, in a case of narrow passages the motion planning algorithm may not return a viable path. Finally, with poorly chosen Motion Primitives the robot will not reach the goal moving backward and forward with crossing but not reaching the goal.

4. Rapidly Explored Random Tree

Path planning is divided into two steps. First the tree with specified number of nodes k_{tree} is created. One node at a time is established as it is presented in figure 5 (algorithm 1). Each node represents position and orientation of the robot in environment and the edge represent motion between two adjacent nodes (between two positions). Position is represented as the vector r which contains x and y coordinate in 2D plane. After building the tree, an optimal path is established by finding the k–nearest nodes that satisfy the distance criterion given by Euclidean distance measured between q_{goal} and selected node being nearest than ε. In our algorithm we assumed that $\varepsilon = 10cm$. From set of all nodes selected in previous step, the one which has smallest execution time is selected. This time is computed as a sum of motions (edges) from any given node to the starting node q_{start}. All motion execution times are presented in table 1.

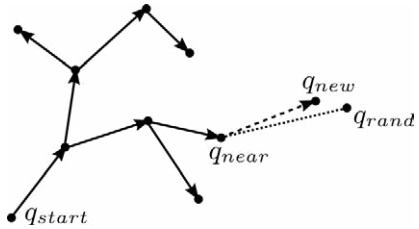

Fig. 5. Building tree T rooted in q_{start} by selecting q_{near} and adding new node in q_{new}

Algorithm 1 Algorithm for building tree T

procedure BUILDTREE
  Input: initial configuration q_{start}, number of nodes k_{tree}, region free from obstacles Q_{free}, set of allowable motions $\Delta q_{motions}$
  Output: Tree T
  if $q_{start} \in Q_{free}$ **then**
    $T.init(q_{start})$
  else
    return INIT_ERROR
  repeat
    $q_{rand} \leftarrow GET\_RAND\_NODE(Q_{free})$
    $q_{near} \leftarrow NEAREST\_NODE(q_{rand}, T)$
    $q_{new} \leftarrow GET\_NEXT\_MOTION(Q_{free}, q_{near}, \Delta q_{motions})$
    if $q_{new} \in Q_{free}$ **then**
      T.add_vertex(q_{new})
      T.add_edge(q_{near}, q_{new})
      $T.update\_tree\_info()$
  until T.number_of_nodes() $< k_{tree}$
  return T

The core pat of RRT algorithm is selection of three nodes: q_{rand}, q_{near} and q_{new}. When finding q_{rand} node two approaches were used: biased and unbiased. Biased approach differs from unbiased by returning q_{goal} node once each k_{bias}–times function is stimulated. For large workspace this can be useful especially near the end of path search (when generated tree is few steps away from the goal configuration). We define configuration as the vector q which contains position and orientation of the robot. The bias reduces unnecessary random points in large workspace and focuses on goal configuration.

For selecting q_{near} node two different approaches were used: Euclidean distance norm and A*–like heuristics.

Euclidean distance between two configurations q_A and q_B is represented by:

$$eucl\_norm(q_A, q_B) = ||r_B - r_A|| \qquad (1)$$

where r_A and r_B are configuration positions. The are described by coordinates of the point located between two feet. This localisation is predefined and is specific for each motion.

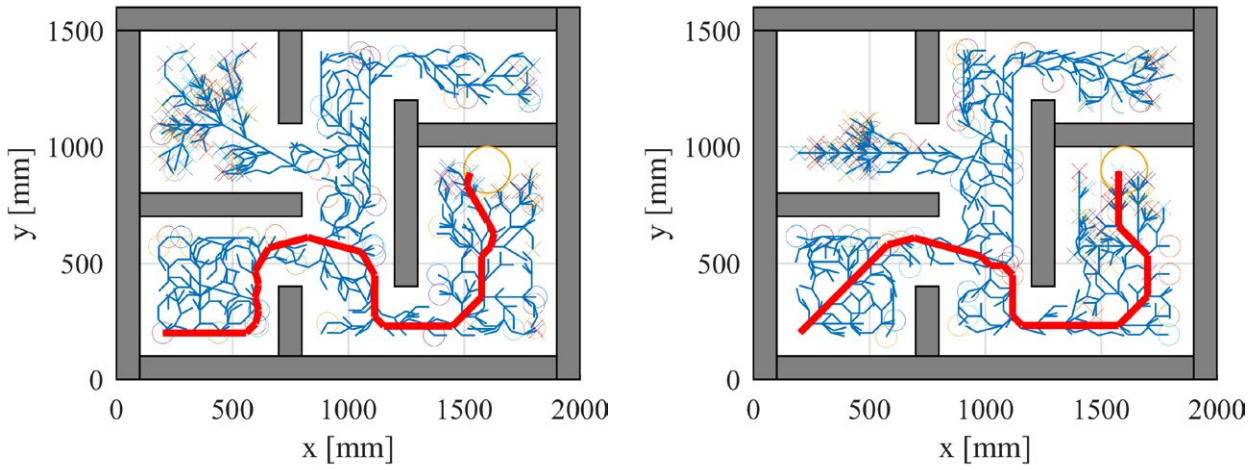

Fig. 6. Example of generated tree with selected path for informed RRT algorithm. Left figure present tree without q_{goal} bias, while right presents the results obtained using the bias

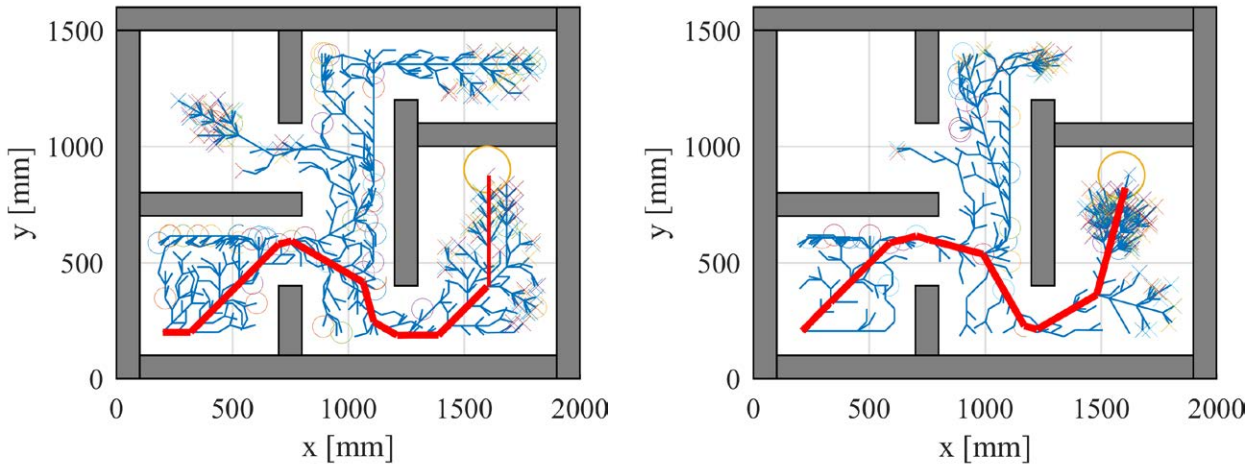

Fig. 7. Example of generated tree path obtained using informed RRT algorithm with A* heuristics. Left figure presents tree without q_{goal} bias, while right presents it with the bias

For implementing A*–like heuristics the standard form of A* algorithm is used:

$$f(q) = g(q) + h(q) \tag{2}$$

In this equation $g(q)$ represents the weight of the path between q_{start} and q. It is the time needed to execute the path from start to the node representing configuration q. $h(q)$ is the weight (in terms of time) for the path segment from the given node describing configuration q to the final configuration q_{end}. In our approach the weight $h(q)$ is obtained using the following formula:

$$h(q) = eucl\_norm(q, q_{end})/(k_{mean} * V_{mean_{all}}) \tag{3}$$

where $V_{mean_{all}}$ is the mean velocity of the robot along the executed path from q_{start} to the node representing configuration q. k_{mean} is the weight parameter used in $f(q)$ function.

5. Experiments

Presented algorithm was tested in the real scenario. Figure 4 (right side) presentes the testbed for the environment with the obstacles. Figure 4 (left side) is presenting automatically generated map obtained by vision system with ArUco markers [5]. In our application ArUco markers are used to localize the obstacles and goal destination. The coordinates of the point S are $[210mm, 205mm]$. Goal destinations are: A, B, C locations $A = [521mm, 1404mm]$, $B = [1589mm, 876mm]$, $C = [1710mm, 1315mm]$. Radius of acceptable tolerance towards the goal position is $r_{goal} = 10cm$.

List of parameters used for generating Motion Primitives is given in Table 1. Using these parameters MGS generates the set of Motion Primitives used in path planning algorithm. Different motions are performed with different speed. Comparing V_{mean} motion No.1 and No.3 from table 1 it can be seen that walking straight is faster than walking diagonally. It im-

Tab. 2. Results of path time execution for analysed algorithm with specified parameters

q_{goal}	metrics	k_{bias}	t_{min}	t_{mean}	t_{max}	n_{min}	n_{mean}	n_{max}
A	euclidean	0%	534.5	733.22	1032	131	279	620
A	euclidean	25%	528	689.23	955	58	203	613
A	A*	0%	522	658.75	834.5	95	442	736
A	A*	25%	478	618.55	788	88	342	707
B	euclidean	0%	504	706.24	927.5	66	181	453
B	euclidean	25%	489.5	674	818	69	155	379
B	A*	0%	461.5	641.7	856	107	256	558
B	A*	25%	484.5	627.7	831	78	228	432
C	euclidean	0%	501	673.4	954	75	168	344
C	euclidean	25%	491	659.5	808	44	60	100
C	A*	0%	473	625.5	773	85	194	391
C	A*	25%	479.5	616.3	735	48	68	134

plies that "zig–zagging" is more time consuming.

Figure 6 presents the example of generated tree with selected path for informed RRT algorithm with (rigth picture) and without (left picture) the bias. In presented tree, nodes excluded from the search are marked by crosses and circles. Cross marks the nodes excluded by time limit, while circle marks nodes with dead—ends.

Figure 7 presents example of generated tree with selected path using informed RRT algorithm with A* heuristics. Right picture presents version with bias while left without it. The reduced amount of nodes in the areas which are farther from the goal configuration can be observed. Figure 6 and figure 7 provide the visible difference in the path curvature. Path in figure 7 utilizes faster straight motions.

The experiments are summarized in table 2. Each test (each row) was performed for $k_{tree} = 800$ 100 times. Goal destinations q_{goal} are represented in figure 4 (left picture) as the A,B,C points.

Analyzing the data given in 2 it can be concluded that the proposed algorithm provides significant reduction of t_{min} and t_{mean} for generated path. In majority of the cases t_{max} obtained for RRT–A* with bias was satisfactory. The values of n_{min}, n_{mean} and n_{max} inform with what amount of nodes the connection with the goal was achieved. The Table is also presenting what k_{tree} should be selected for successful obtaining of the path from the starting to final position. For all tests $k_{tree} = 800$ provided the success. While RRT–A* reducs significantly t_{mean} it requirs bigger number of nodes in generated tree. This can lead to unsuccessful path planing result when the indicated path is not connecting the initial and final node. Further analysis and improvements should focus on this aspect.

6. Conclusion

This article describes the humanoid path planning algorithm for cluttered environment. The pre–generated Motion Primitives are used. The incorporation of of path planning algorithm to the control structure of is shortly discussed. Key data of four algorithms are given and analyzed. Basis on it, it was concluded that the presented RRT algorithm with A* heuristics reduces the time needed for executing the path. The bias function used for generating q_{rand} nodes reduces the size of the tree used for finding the path (n_{min}). The further work will be considering the variable robot orientation in the path planning. It is obviously useful for the robots performing the range of tasks. Additionally, the set of movements for turning the robot on the spot should be generated.

Although presented algorithm provides viable path for considered scenario, it is not free of limitations. First of all, with limited number of Motion Primitives (such as in our paper) it is possible that the viable path will be not delivered e.g. a narrow corridor case. Secondly, with proposed set of Motion Primitives it is not possible to refine the path as it can be done in RRT* algorithm versions. This may be overcome either by on-line generation of the Motion Primitives or by introducing more Motion Primitives. Finally, a combination of Motion Primitives covering the short and long distances should be considered. "Short length" Motion Primitives provide possibility for fine adjustment of the robot position or deliver the motion in narrow passages, while the "long length" Motion Primitives should be preferred globally (in open spaces) what decreases the computational effort.

AUTHORS

Maksymilian Szumowski* – Warsaw University of Technology, Institute of Aeronautics and Applied Mechanics, Nowowiejska 24, Warsaw, 00–665, e-mail: mszumowski@meil.pw.edu.pl, www: https://ztmir.meil.pw.edu.pl/web/Pracownicy/inz.-Maksymilian-Szumowski.

Teresa Zielinska – Warsaw University of Technology, Institute of Aeronautics and Applied Mechanics, Nowowiejska 24, Warsaw, 00–665, e-mail: teresaz@meil.pw.edu.pl, www: https://ztmir.meil.pw.edu.pl/web/Pracownicy/prof.-

Teresa-Zielinska.

*Corresponding author

ACKNOWLEDGEMENTS

The project was funded by POB Research Centre for Artificial Intelligence and Robotics of Warsaw University of Technology within the Excellence Initiative Program - Research University (ID-UB).

REFERENCES

[1] J. Baltes, J. Bagot, S. Sadeghnejad, J. Anderson, and C.-H. Hsu, "Full-body motion planning for humanoid robots using rapidly exploring random trees", *KI - Künstliche Intelligenz*, vol. 30, no. 3-4, 2016, 245–255, 10.1007/s13218-016-0450-z.

[2] J. L. Blanco, M. Bellone, and A. Gimenez-Fernandez, "TP-space RRT – kinematic path planning of non-holonomic any-shape vehicles", *International Journal of Advanced Robotic Systems*, vol. 12, no. 5, 2015, 55, 10.5772/60463.

[3] W. Chi and M. Q.-H. Meng, "Risk-RRT∗: A robot motion planning algorithm for the human robot coexisting environment". In: *2017 18th International Conference on Advanced Robotics (ICAR)*, 2017, 10.1109/icar.2017.8023670.

[4] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT∗: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, 10.1109/iros.2014.6942976.

[5] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion", *Pattern Recognition*, vol. 47, no. 6, 2014, 2280–2292, 10.1016/j.patcog.2014.01.005.

[6] E. Krotkov, D. Hackett, L. Jackel, M. Perschbacher, J. Pippine, J. Strauss, G. Pratt, and C. Orlowski, "The DARPA robotics challenge finals: Results and perspectives", *Journal of Field Robotics*, vol. 34, no. 2, 2016, 229–240, 10.1002/rob.21683.

[7] S. M. LaValle, "Rapidly-exploring random trees : a new tool for path planning". In: *Mathematics*, 1998.

[8] J. Li, S. Liu, B. Zhang, and X. Zhao, "RRT-a∗ motion planning algorithm for non-holonomic mobile robot". In: *2014 Proceedings of the SICE Annual Conference (SICE)*, 2014, 10.1109/sice.2014.6935304.

[9] T. Nishi and T. Sugihara, "Motion planning of a humanoid robot in a complex environment using RRT and spatiotemporal post-processing techniques", *International Journal of Humanoid Robotics*, vol. 11, no. 02, 2014, 1441003, 10.1142/s0219843614410035.

[10] M. Paetzel and L. Hofer, "The RoboCup humanoid league on the road to 2050 [competitions]", *IEEE Robotics & Automation Magazine*, vol. 26, no. 4, 2019, 14–16, 10.1109/mra.2019.2945738.

[11] Z. Qiu, A. Escande, A. Micaelli, and T. Robert, "A hierarchical framework for realizing dynamically-stable motions of humanoid robot in obstacle-cluttered environments". In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, 2012, 10.1109/humanoids.2012.6651622.

[12] E. Szadeczky-Kardoss and B. Kiss, "Extension of the rapidly exploring random tree algorithm with key configurations for nonholonomic motion planning". In: *2006 IEEE International Conference on Mechatronics*, 2006, 10.1109/icmech.2006.252554.

[13] M. Szumowski and T. Zielinska, "Preview control applied for humanoid robot motion generation", *Archives of Control Sciences*, 2019, 10.24425/ACS.2019.127526.

[14] M. Szumowski, M. S. Żurawska, and T. Zielińska, "Simplified method for humanoid robot gait generation". In: *Advances in Mechanism and Machine Science*, 2019, 2269–2278, 10.1007/978-3-030-20131-9_224.

[15] Z. Tahir, A. H. Qureshi, Y. Ayaz, and R. Nawaz, "Potentially guided bidirectionalized RRT∗: for fast optimal path planning in cluttered environments", *Robotics and Autonomous Systems*, vol. 108, 2018, 13–27, 10.1016/j.robot.2018.06.013.

[16] M. Vukobratović and B. Borovac, "ZERO-MOMENT POINT — THIRTY FIVE YEARS OF ITS LIFE", *International Journal of Humanoid Robotics*, vol. 01, no. 01, 2004, 157–173, 10.1142/s0219843604000083.

[17] K. Yang, S. Moon, S. Yoo, J. Kang, N. L. Doh, H. B. Kim, and S. Joo, "Spline-based RRT path planner for non-holonomic robots", *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, 2013, 763–782, 10.1007/s10846-013-9963-y.