11

Navigation Planning with Human-Like Approach

Yasar Ayaz*, Atsushi Konno*, Khalid Munawar**,
Teppei Tsujita* and Masaru Uchiyama*
*Tohoku University, **National University of Sciences and Technology (NUST)
*Japan, **Pakistan

1. Introduction

Moving about in everyday human environments such as homes and offices would require a robot to cope with terrains not ordinarily traversable by robots on wheels. These may include stairs and stepping stones etc. This is because our homes and offices are custom designed for biped walkers i.e. humans. The usage of legs rather than wheels for mobility was thus an inevitable evolution in the history of mobile robotics. Giving legs to a robot instead of wheels gives it a lot more than just resemblance to a human being. Unlike ordinary mobile robots, humanoids have the ability to cross obstacles by stepping over or upon them. This ability is left unexploited if the algorithms used for ordinary mobile robot navigation among obstacles are employed for humanoids too.

Various approaches have been adopted in order to address this problem in the past. (McGhee & Iswandhi, 1979) developed a method that divides the terrain into 'permissible' and 'impermissible' stepping positions. Keeping in view the direction in which the ultimate goal position is located, the robot selects the next foothold from amongst the permissible ones in the immediately reachable footholds while taking care of postural stability constraints. While this method targeted a general application to legged robotics, (Yagi & Lumelsky, 1999) presented another one that deals specifically with humanoid robots. Depending upon the distance from the obstacle nearest to the robot in the direction of motion (presumably the direction in which the goal position is located) the robot adjusts the length of its steps until it reaches the obstacle. Now, if the obstacle is small in size, it is overcome by stepping over or upon it whereas if it is too tall to be overcome in this way, the robot starts sidestepping until it moves clear of the obstacle. Obviously, whether to sidestep left or right is also a pre-programmed decision. These and other such localized reactive approaches have the tendency to lead the robot into a local loop or a deadlock in which case the robot would have to be backtracked in order to follow an alternate path.

(Kuffner et al., 2001) argued that since such reactive algorithms failed to consider complexities occurring in the path at a later stage before opting to take it, they ended up stuck in local loops and deadlocks. In order to tackle this problem they presented a footstep planning algorithm based upon game theory that takes into account global positioning of obstacles in the environment. This technique has been tested on H6 (Kuffner et al., 2001), H7

(Kuffner et al., 2003), Asimo Honda (Chestnutt et al., 2005; Michel et al., 2005) and HRP-2 (Michel et al., 2006) humanoid robots with some improvements. The algorithm uses a discrete set of predefined stepping locations in the robot's immediate vicinity for either leg. Also predefined are intermediate postures that the robot assumes while moving its feet between any two of these stepping locations. Selecting a set of these possible stepping locations, each of which leads to a similar set of descendants, a tree is spread out from the initial footstep position. Now, after pruning from the tree those branches that do not end up with a step in the destination area, a two-levelled polygon-polygon collision search is conducted in order to identify trajectories that result in collision with the obstacles in the environment. Once these are discarded too, a greedy search is conducted in order to hunt out the best among the paths generated. This strategy cannot distinguish between the paths without obstacles and those with some obstacles that can be stepped over due to which the robot always needs to consider high stepping profiles in every step. In addition, because of generation of a large search tree of possible stepping locations, greedy search has to be applied instead of an exhaustive one. On the other hand, a human in such a situation would start by searching for an obstacle-free path directly connecting initial and goal locations. If found, it would simply be adopted. Whereas some options would be considered once an obstacle-free direct path is not found. Again, an obstacle can be dodged from the right, or dodged from the left, or is crossed by stepping over.

Our algorithm starts by checking if it is possible for the robot to go directly from the initial to the final location. If so, such a path is adopted. If not, trajectories are formed in order to dodge the hindering obstacle from the right and the left. In addition, keeping in view the dimensional constraints, obstacles are classified into three types. The smallest ones that can be stepped over: type-1. For these an additional trajectory considering stepping over is also formed. The larger ones that cannot be stepped over but the robot can pass right next to them: type-2. And the largest ones that might collide with the robot's arms (sticking out wider than its feet) if the robot passes right next to them: type-3. The robot keeps a larger distance from obstacles of type-3 as compared to those of type-2 in order to avoid collision. In this way, the algorithm starts by considering the shortest path and then expands its alternatives on either side spreading out considering the obstacle type and proximity. In addition, by identifying obstacles once encountered, it avoids getting stuck into the local loops and deadlocks. It is noticeable here that branches formed from every node in the search tree can be a maximum of 3, and for several straight line segments each node leads to only one child. This reduces the branching factor making a smallest possible search tree which in turn enables us to apply the exhaustive search for seeking the best possible path. It can immediately be noticed that this search, though exhaustive, is computationally cheap.

This article explains our proposed method in detail. In section 2 we examine some related research. Section 3 introduces the concept of human-like footstep planning and highlights salient differences with the existing approaches. The footstep planner is explained in detail in section 4 along with simulation results using a model of HRP-2 humanoid robot. Section 5 states the conclusion and some ideas for future work.

2. Game Theory Based Footstep Planning

Kuffner's algorithm for footstep planning among obstacles for biped robots (Kuffner et al., 2001; Kuffner et al., 2005) is a global motion planning strategy based on using game theory for finding alternate paths in an obstacle cluttered environment by considering a discrete set of heuristically chosen statically stable footstep locations for a humanoid robot and identifying the better path via cost heuristic implementation using greedy search. The assumptions made by the algorithm are:

(i). The environment floor is flat and cluttered with non-moving obstacles of known position and height,
(ii). A discrete set of feasible, statically-stable footstep placement positions and associated stepping trajectories are pre-computed,
(iii). Only the floor surface is currently allowed for foot placement (not obstacle surfaces).

A static stability analysis is carried out for the stepping motion of the humanoid robot as a result of which a continuous region is identified where the robot can place its lifted foot (while balanced on one leg) without losing stability. Since it is computationally extensive to check every single point in this region for a footstep placement possibility in every step, a discrete set of 'feasible' footstep locations is heuristically identified in this region. Standing balanced on one foot, the robot would alternatively consider placing its foot at each of these locations and from each stepping possibility more nodes are generated in a similar manner. Foot transition from each location to the next is carried out via heuristically identified statically stable intermediate postures Q_{right} and Q_{left} (for standing balanced on right and left foot respectively) in order to limit the number of possible transition trajectories considered in every step. This is followed by best path identification through greedy search by employing a heuristic cost function which attempts to approximate an exhaustive search while applying a greedy one since exhaustive search itself would be very time consumptive given the large search tree of possible stepping locations the algorithm generates (Kuffner et al., 2001; Kuffner et al., 2005).

The algorithm was simulated on a model of H6 humanoid robot (Kuffner et al., 2001; Kuffner et al., 2005). Considering 15 discrete footstep placement options in each step in a room with 20 obstacles, it formed search trees of 6,700 and 830,000 nodes in (Kuffner et al., 2001) and (Kuffner et al., 2005) respectively. Best path was traced out using greedy (Kuffner et al., 2001) and A* (Kuffner et al., 2005) search strategies with total processing time of 12 s on 800 MHz Pentium-II and 4 s on 1.6 GHz Pentium-IV running Linux respectively.

The main drawback of this algorithm is high computational complexity because of generation of a large search tree of possible stepping locations which reflects in terms of time consumption. Moreover, a closer look reveals that since this algorithm is not reactive in nature, it does not distinguish between steps taken to cross over obstacles and those taken ordinarily. Since the same intermediate postures as in ordinary steps (Q_{right} and Q_{left}) have to provide the robot with trajectories capable of stepping over obstacles too, they involve lifting of the foot to a high position as if stepping over obstacles in every step. Yet, since the

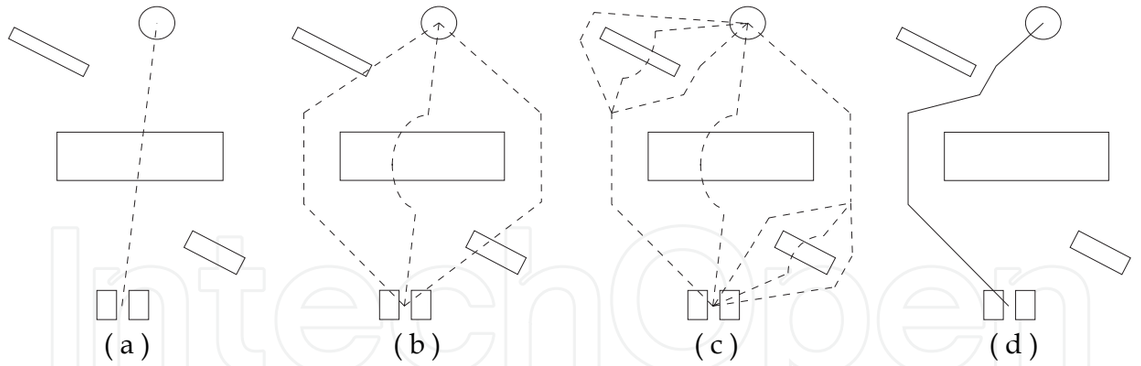


Fig. 1. An illustration of the human-like footstep planning strategy.

tree of step options in game theory based footstep planning strategy is spawned without paying heed to presence or absence of obstacles in the environment, stepping over is merely a result of obstacles fortunately found lying between present and next footstep locations without causing a collision during the stepping trajectory. In addition, when it comes to searching for the best path amongst those traced out by the algorithm, an exhaustive search cannot be applied because of the large tree of possible stepping locations generated (Kuffner et al., 2001; Kuffner et al., 2005). A greedy search, however, might not hunt out the best solution under many circumstances (Cormen et al., 1994). In order to tackle some of these problems a tiered planning strategy has been introduced that splits the planner into high, middle and low level layers in order to traverse different terrain types (Chestnutt & Kuffner, 2004).

3. Human-Like Footstep Planning

3.1 Graph Expansion Strategy

As humans we know that, unlike localized reactive navigation strategies, paths adopted by us to traverse obstacle cluttered terrains are planned while taking into account global positioning of obstacles around us. Also, humans do not conduct navigation planning by random spreading of footstep placement possibilities throughout the terrain while ignoring the relative position of obstacles in the environment and later evaluating which of these would result in collision as is done by the game theory based approaches. Quite differently, humans start by first searching for the path directly connecting the start and destination points. If found available, this path is simply taken without considering alternatives. On the other hand, if the direct path is found hindered by an obstacle, we consider paths dodging it as well as stepping over it with the better one of these being adopted eventually.

This is precisely the approach we seek to adopt. In our method the planner starts by looking for the presence of obstacles in the path directly connecting the initial and goal locations. If available, this path is taken without heed to alternate possibilities. If unavailable, paths are generated from both right and left of the nearest obstacle hindering the direct passage. In addition, if the obstacle is small enough, a path is also planned by stepping over it. If another obstacle is found hindering a generated alternate path, paths from right, left and over this obstacle are also analyzed in a similar fashion. In this way the graph evolves from simple to complex paths.

Fig. 1 elaborates upon graph expansion in a simple scenario. Intended destination is marked by a circle. Checking the direct path from initial to final position (Fig. 1 (a)), an obstacle is found hindering the way. Pivot points are chosen near the corners of this obstacle in order to find paths to dodge it and a path is also planned by stepping over (Fig. 1 (b)). Even sections of these alternate paths formed are found hindered by other obstacles, following which, paths are designed by circumvention and stepping over (Fig. 1 (c)). Based on heuristic cost assignment (depending upon step complexity in case of humanoids), exhaustive search for best path gives us the desired solution (Fig. 1 (d)).

Keeping in view the dimensional constraints, obstacles are classified into three types. The smallest ones that can be stepped over: type 1. For these a trajectory considering stepping over is also formed. The larger ones that cannot be stepped over but the robot can pass right next to them: type 2. And the largest ones that might collide with the robot's arms (sticking out wider than its feet) if it passes right next to them: type 3. The robot keeps a larger distance from obstacles of type 3 as compared to those of type 2 in order to avoid collision. In this way, the algorithm starts by considering the shortest path and then expands its alternatives on either side, spreading out considering the obstacle type and proximity. In addition, by identifying obstacles once encountered, it avoids getting stuck into the local loops and deadlocks. It is noticeable here that branches formed from every node in the search tree can be a maximum of three, and for several straight line segments each node leads to only one child. This reduces the branching factor making a smallest possible search tree which in turn enables us to apply exhaustive search for seeking the best possible path. It can immediately be noticed that this search, though exhaustive, is computationally cheap.

3.2 Comparison with Visibility Graph

Due to the fact that our algorithm uses straight line footstep sequences to approach pivot points (located near obstacle edges) on which turning is carried out, some readers may confuse our method with the well-known visibility graph approach used for mobile robot motion planning (Latombe, 1991). Therefore, it is important to highlight salient differences between the two approaches.

Of course one clear difference is that we also plan paths for crossing obstacles by stepping over, which the visibility graph algorithm does not. However, more fundamental differences in the graph, as well as in the way that it is expanded, reveal themselves upon a keener examination. For the analysis given in this subsection, we limit our graph expansion methodology to only dodging obstacles by circumvention in order to be able to readily make comparison with the visibility graph approach.

3.2.1 Graph Planning

A visibility graph is a graph of *intervisible* locations. Each obstacle corner in the environment is included in the graph and represents a vertex (or pivot point) about which turning is possible. A path is a visible (i.e. collision free) connection between any two obstacle corners.

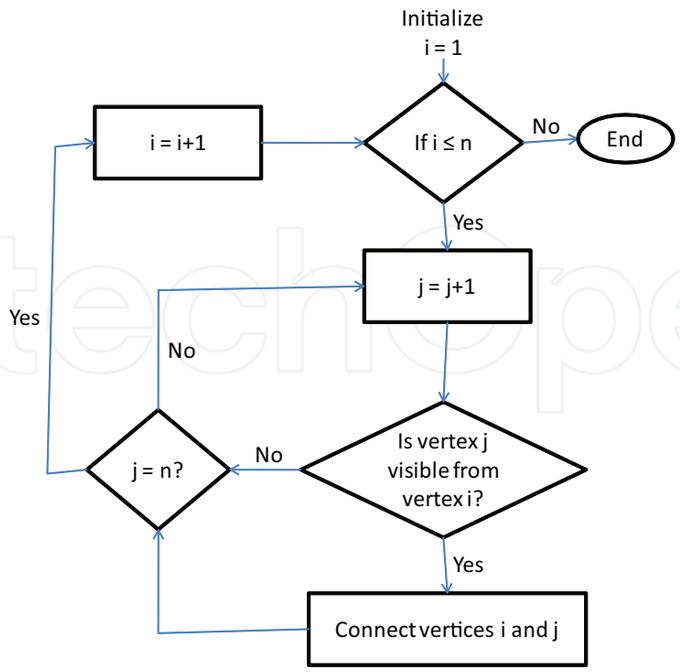


Fig. 2. A flowchart explaining the visibility graph expansion process.

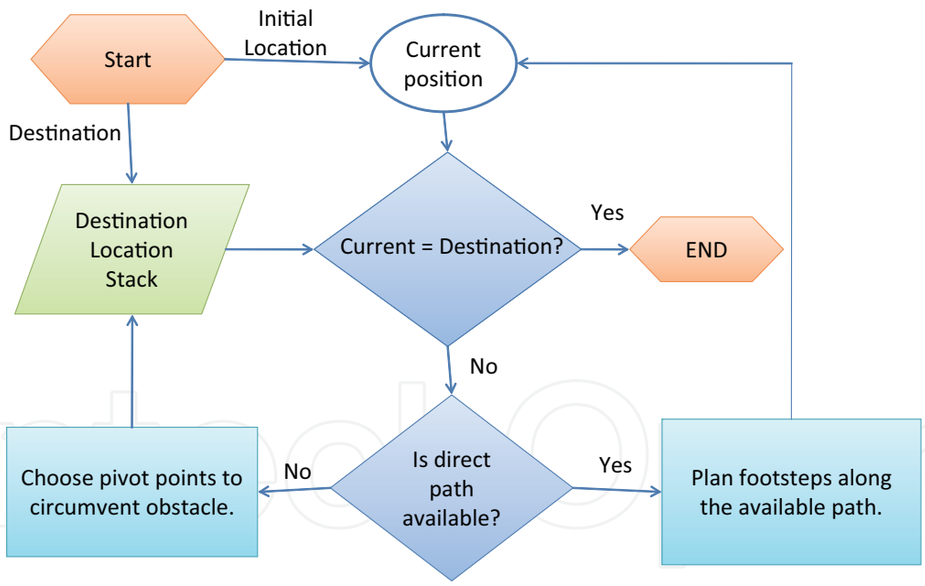
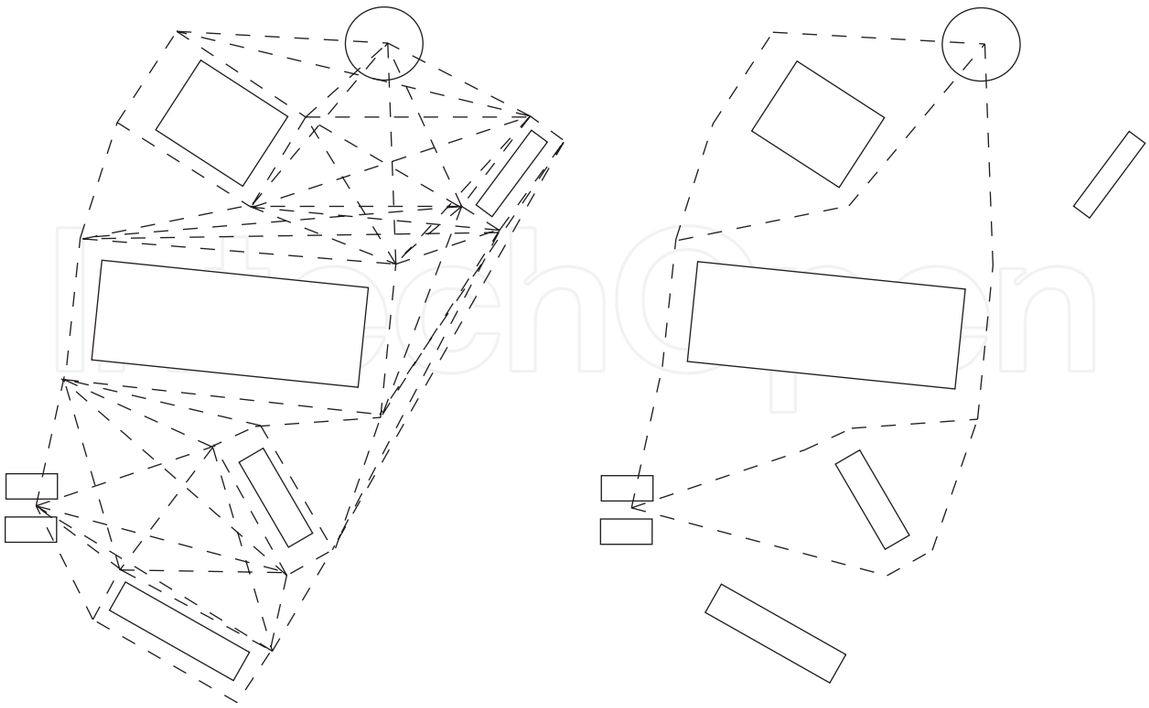


Fig. 3. A flowchart describing the graph expansion process using our planning strategy. Here, the graph expansion is limited to path planning by circumvention only.

If two obstacle corners can see each other (i.e. if a collision free direct path is possible in order to travel between them), they are connected by a possible path.

Fig. 2 illustrates the visibility graph expansion process with the help of a flowchart. The way to expand a visibility graph is to simply include all the obstacle corners into the graph as vertices (or pivot points) and interconnecting them by all possible collision free direct paths.



(a) The visibility graph expanded. (b) Graph expanded using our approach.

Fig. 4. A comparison between the visibility graph and the graph expanded using our method.

On the contrary, our approach is evolutionary and the graph is expanded from simpler to more complex paths. First, only the direct path interconnecting the start and destination points is checked for the presence of obstacles. If it is available, it is simply taken and no other paths are generated. If it is not available, paths are generated from the right and left to dodge the nearest obstacle hindering the direct path. If any of these alternate paths are also found hindered by an obstacle, alternate paths are generated to dodge this obstacle from right and left too in a similar manner. A flow diagram of graph expansion using our approach is given at Fig. 3. Please note that in the flow diagram at Fig. 3 as well as in the above description we are not referring to the paths formed by stepping over obstacles in order to restrict graph expansion to planning by circumvention only for the sake of comparison with visibility graph.

The difference between the two algorithms is very clear in terms of the manner of graph expansion. A visibility graph is directionless and simply includes all the obstacle edges into the graph. Our algorithm, on the other hand, is of an evolutionary nature. It expands the graph keeping in view the direction of motion and increases the number of vertices or paths only if a direct path to an intermediate or ultimate destination point is found hindered by an obstacle.

The graph expansion approach in the two algorithms, therefore, is significantly different.

3.2.2 Graph Output

It is insightful to look at the outputs produced by both the algorithms in order to observe their comparative uniqueness.

Fig. 4 shows a robot in an obstacle cluttered environment with the destination marked by a circle. Fig. 4 (a) shows a visibility graph expanded in this scenario whereas Fig. 4 (b) shows the graph expanded using our approach. As it can clearly be seen, our algorithm produces a much simpler graph comprising only the meaningful paths. This is because of the evolutionary approach inspired by human navigation that we adopt for path planning.

4. Planner Design and Simulation

4.1 Kinematics and Stability Analysis

For initial simulation work, our algorithm inherits assumptions (i) and (iii) from the game theory based method described in section 2. Also, as in assumption (ii) we also employ a static stability based criteria for stepping motions in the simulation phase.

Selection of footstep locations for ordinary walking is done according to the intended direction of motion. Right or left foot respectively is used to take the first step depending upon whether the destination is located to the right or left of the robot's current position. For ordinary steps a half-sitting posture is used. Maximum possible step-length (1.9 cm approximately for HRP-2) is used in each step. Final step when reaching the destination is shortened to step exactly within the destination area.

Backward steps are not considered while planning since back-stepping is one of the very situations a footstep planner is primarily meant to avoid (Ayaz et al., 2006; Ayaz et al., 2007). Thus, kinematically reachable area in the forward direction only (highlighted in Fig. 5 for the right foot) is used for stepping.

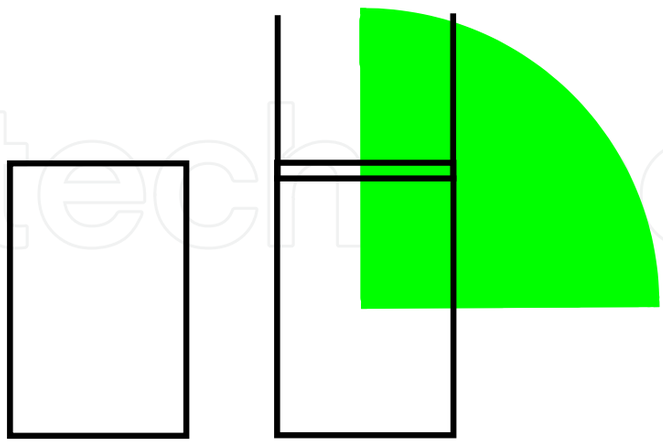


Fig. 5. Stepping area for normal half sitting posture.

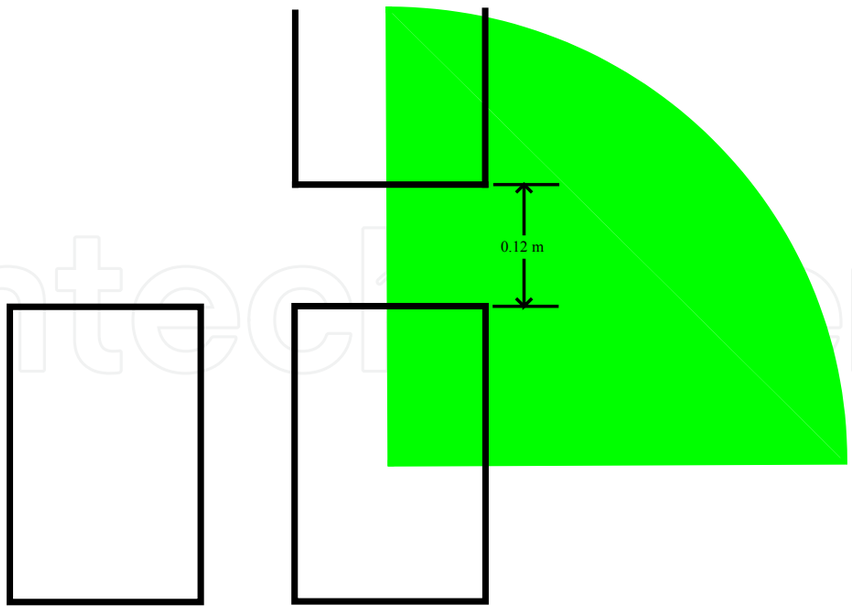


Fig. 6. Creating space between current and next stepping locations by lowering the waist.

As it can be seen from Fig. 5, even stretching the foot to its maximum is not sufficient enough to create space between current and next step positions. In other words, this posture for normal stepping cannot be used to step over obstacles as well. Thus for stepping over obstacles, the knees of the robot are bent further and the body is lowered in order in order to increase the possible step-length. In this way a gallery of 11.9 cm is created between current and next stepping locations in which the obstacle could be located while it is stepped over (illustrated in Fig. 6). The trajectory of the foot used is similar to that employed for our earlier work on Saika-3 humanoid robot (Ayaz et al, 2007). The height of HRP-2's ankle joint is approximately 11 cm. Obstacles of height ≤ 10 cm and depth ≤ 11 cm are thus considered possible to be stepped over for the sake of this simulation work. It is to be borne in mind that here the dimensional constraints are determined neither while accounting for dynamic stability of the robot nor are limiting in terms of the maximum obstacle height possible to be overcome using HRP-2.

4.2 Obstacle Classification

In human environments, obstacles are of a variety of shapes and sizes. Our algorithm requires identification of a box-like boundary around each obstacle such that the obstacle of arbitrary shape is entirely contained inside it. This box-like boundary must be a four cornered shape when viewed from the top but there is no restriction on it being a rectangle or a parallelogram as long as it is a four cornered shape in top view. However, stepping over is only attempted in case the width and height of the boundary that binds the obstacle satisfy the constraints described in Section 4.1. Our algorithm at this stage considers the entire area inside this boundary, in which the obstacle of arbitrary shape is contained, as an obstacle. These obstacles are classified into three types:

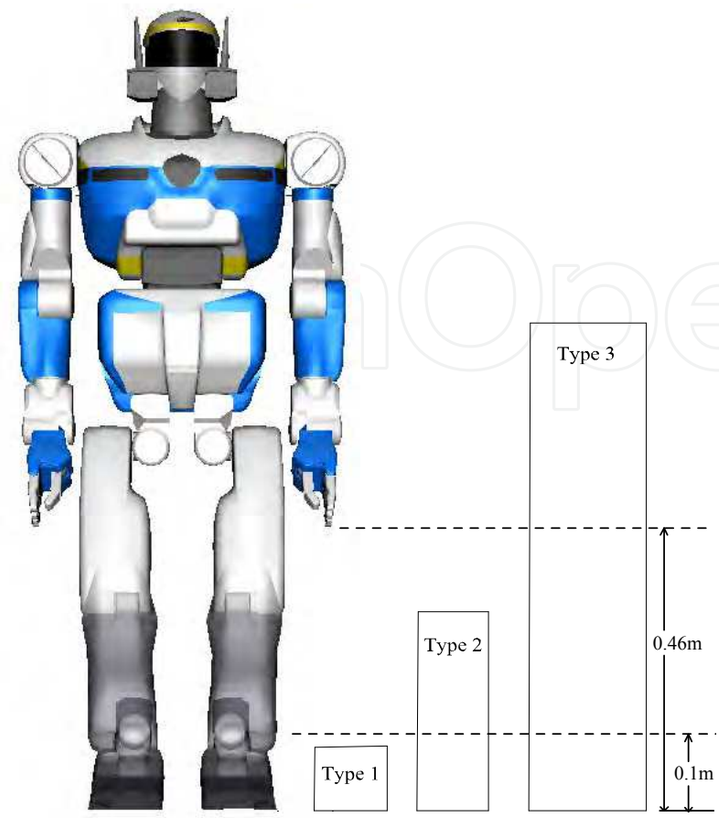


Fig. 7. Obstacles classified into types on the basis of their heights.

Type-1: Height ≤ 0.1 m	Can be crossed by stepping over if depth ≤ 0.11 m.
Type-2: 0.1 m < Height < 0.46 m	Cannot be crossed by stepping over and cannot collide with the robot's arms.
Type-3: Height ≥ 0.46 m	Cannot be crossed by stepping over and while dodging we have to make sure the robot's arms do not collide with it.

4.3 Collision Detection

Extending lines from the current body-corners of the robot to the to-be body-corner locations at the goal position, a gallery is formed. Using two of these boundary-lines and more drawn between corresponding current and destined points, we form a mesh. If any of these line segments is found intersecting a side of an obstacle (marked by a line between two of its neighbouring corners), we say that a collision has been detected. Then, based upon the distance from the intersection point and the angles made from current position, near and far sides and left and right corners of the obstacle are identified respectively.

The collision detection strategy can be understood more easily using Fig. 8. Using the angle of the intended line of motion, an imaginary set of initial footstep locations is generated at

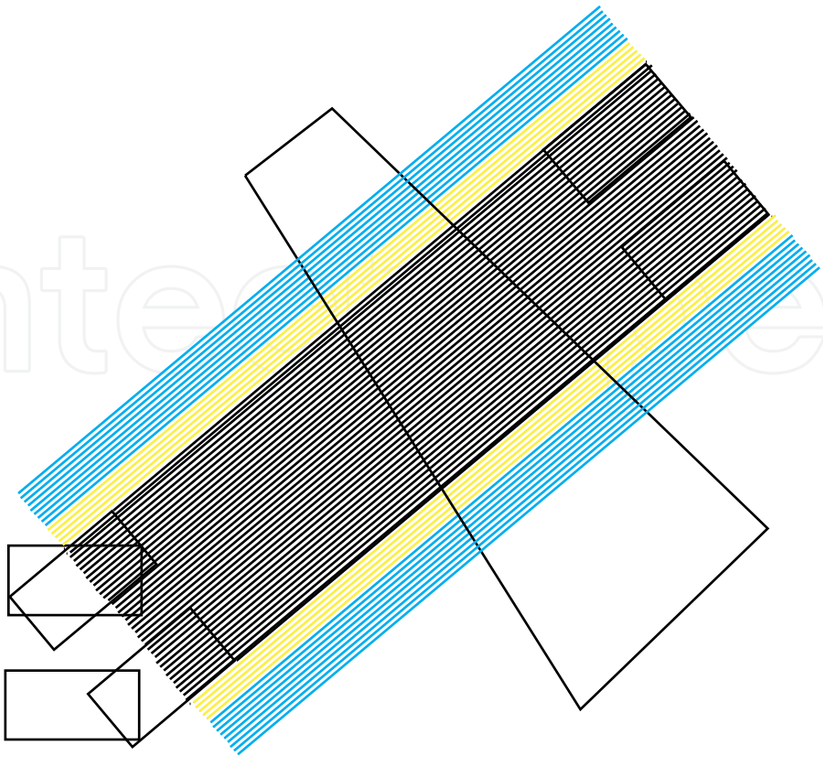


Fig. 8. Mesh of lines drawn for collision detection / prediction.

the current position which indicates the initial position of the robot after it will orientate itself towards the destination. In this direction of motion, another set of imaginary footstep locations is generated at the destination point, which marks the locations at which the robot will place its feet upon reaching the destination. Joining the outer boundaries of these initial and final imaginary footstep locations, a gallery is formed. Inside this gallery, a mesh of equidistant line segments joining corresponding points at the initial and final locations is generated. Each of these lines is checked for intersection with each of the walls of all obstacles in the environment (which are also marked by line segments when looked at from the top). If an intersection is found, a collision is said to have been detected and the robot seeks to form trajectories in order to overcome this obstacle in the path. This type of collision detection is performed not only between the starting point and the ultimate destination, but between every two points which mark the initial and final locations of every intermediate path the robot considers traversing.

From Fig. 8 it can also be seen that a part of the mesh of lines extends beyond the outer boundary of the footholds (drawn in blue and green). These are drawn by joining the outer ends of the arms of the robot (which stick out wider than its feet) at initial and goal locations. Only if an obstacle of type-3 is found to be colliding with a line in the blue part of the mesh, a collision is said to have been detected. The green part, however, checks for both type-2 and 3 obstacles but is insensitive to obstacles of type-1.

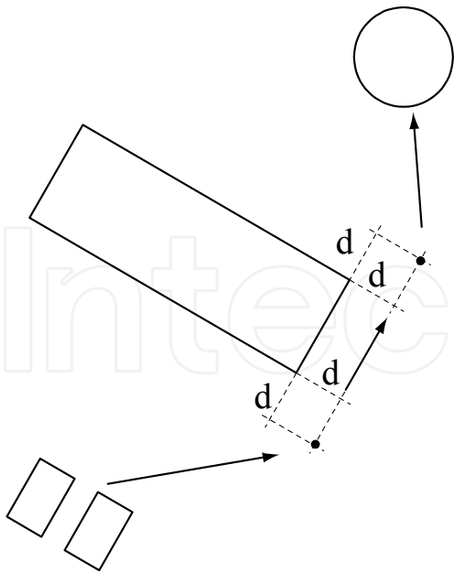


Fig. 9. Chosing pivots to dodge an obstacle.

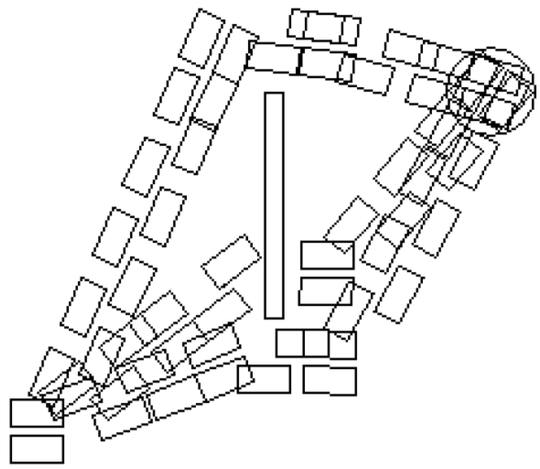


Fig. 10. Overcoming an obstacle of type-1.

For simulation results presented in this chapter, the collision checking mesh used comprises 31 lines with a distance of less than 2 cm between every two neighbouring line segments.

4.4 Overcoming an Obstacle

To dodge an obstacle from a side we choose pivot points near the obstacle corners as shown in Fig. 9.

The distance 'd' along the extended side is chosen such that no part of the robot's body collides with the obstacle as the robot stands in double support state with its body centre at the pivot point. For instance, in case of type-1 obstacles, this distance is equal to half the length of the diagonal of the rectangular support polygon formed when the robot stands with both feet next to each other. This is because the outer most edges of the feet are the points closest to the obstacle with which there might be a chance of collision. 'd' can thus be different for each obstacle type but not for obstacles of one group.

As explained in section 4.3, to step over an obstacle, the robot balances itself on both legs one step short of the closest step-location near the obstacle. Next, based on rightward or leftward tilt of the obstacle in relevance with the robot's trajectory, it places forward left or right foot respectively and balances itself on it. Using enhanced stepping motion, the robot now takes a step forward with its movable leg, making sure that the extended foot lands with its back-side parallel to the obstacle's away-side. Fig. 10 displays possible trajectories generated to overcome an obstacle of type-1 by circumvention and stepping over.

4.5 Local Loops and Deadlocks

Each obstacle in the surroundings is allotted an identification number. The planner keeps a history of the obstacles it has overcome in a path as it plans footsteps. Whenever an obstacle

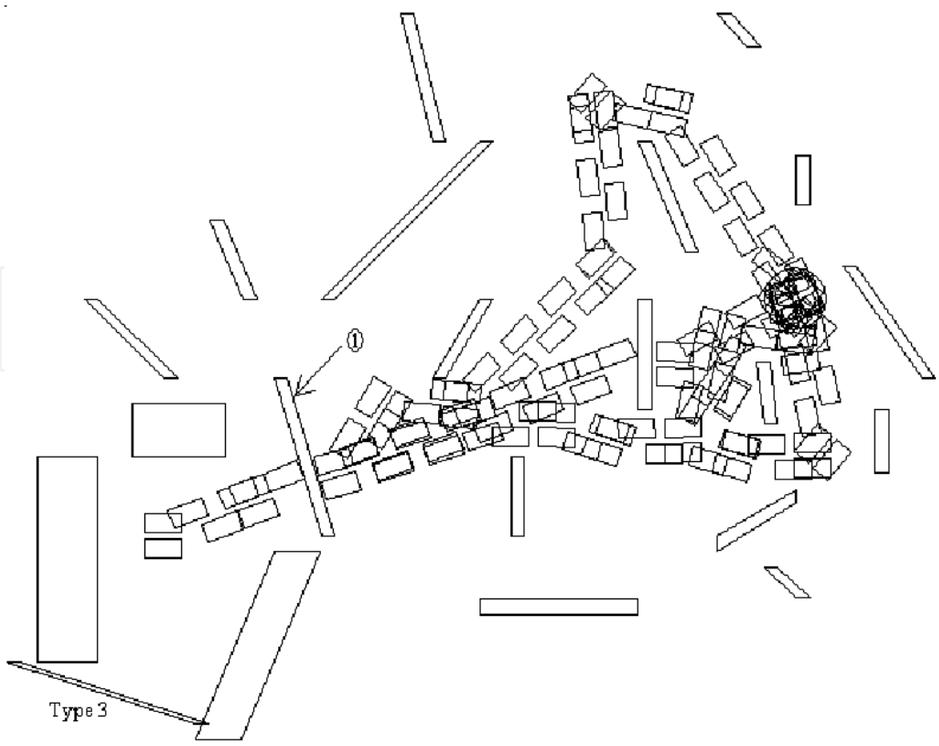


Fig. 11. Avoiding local loops and deadlocks while crossing obstacle '1' of type-1.

occurs twice, it indicates a local loop or deadlock since attempting to overcome it again is bound to bring the planner back to the same position again and again. Such a trajectory is immediately abandoned and pruned from the search tree.

One such situation where the robot encounters a local loop and deadlocks while trying to dodge the obstacle labelled '1' from both right and left sides is shown in Fig. 11. For instance, when trying to dodge the obstacle labelled '1' in Fig. 11 from the right, the robot chooses pivot points to pass from its right side as elaborated upon in section 4.4. However, this path is obstructed by another obstacle on the robot's right. To dodge this newly encountered obstacle, once again the robot forms trajectories from its right and left. The one attempted to pass from left finds the obstacle labelled '1' in the way again. Since this obstacle is present in the history as one that has already been attempted to be dodged, the trajectory for dodging the newly encountered obstacle from the left is discarded as a situation where a deadlock is encountered. The trajectory formed to dodge it from the right, however, finds another obstacle (indicated as a type-3 obstacle in Fig. 11). Attempting to dodge this type-3 obstacle from the left results in a deadlock just as in the previous case whereas the one attempted from its right encounters yet another obstacle. This process is repeated twice more until, in an effort to dodge from the right the obstacle to the left of the obstacle labelled '1', the obstacle labelled '1' is encountered again. This trajectory, therefore, is also abandoned and a local loop is said to have been encountered.

A similar situation occurs while trying to dodge the obstacle labelled '1' in Fig. 11 from its left side. Thus the only trajectory possible to overcome this obstacle which is free of local

loops and deadlocks, is the one formed by stepping over. As we can see, the planner only plans the successful trajectory, avoiding getting stuck in local loops and deadlocks.

4.6 Cost Assignment

A heuristic cost based on the order of complexity of each stepping motion is given at Table 1. These costs are assigned to foot placements as they are planned.

Step type	Description	Cost
Straight	Step placed straight in forward direction	1
Turning	Step placed to change orientation of the robot	2
Extended	Step in which foot turning is combined with extension	3
Enhanced	Step taken for stepping over obstacles	4

Table 1. Heuristic costs assigned to different step types

4.7 Intermediate Paths

If, in order to proceed towards a pivot point while dodging an obstacle from its side, another obstacle is encountered along the way, alternate paths based upon the obstacle type are formed. All these alternate paths converge at the pivot point, meaning that they will all have similar descendants. Thus, the number of these intermediate alternate paths is multiplied by the number of descendent paths and added to the total number of possible alternate paths. Thus, evaluating cost of each intermediate path and keeping only the best during alternate path creation reduces the number of paths to only useful ones.

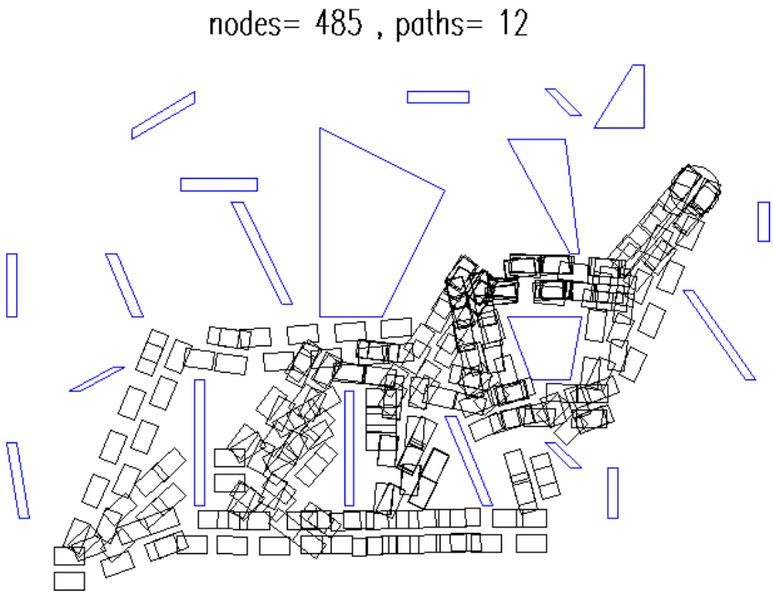


Fig. 12. Various paths for reaching the destination planned for HRP-2 humanoid robot.

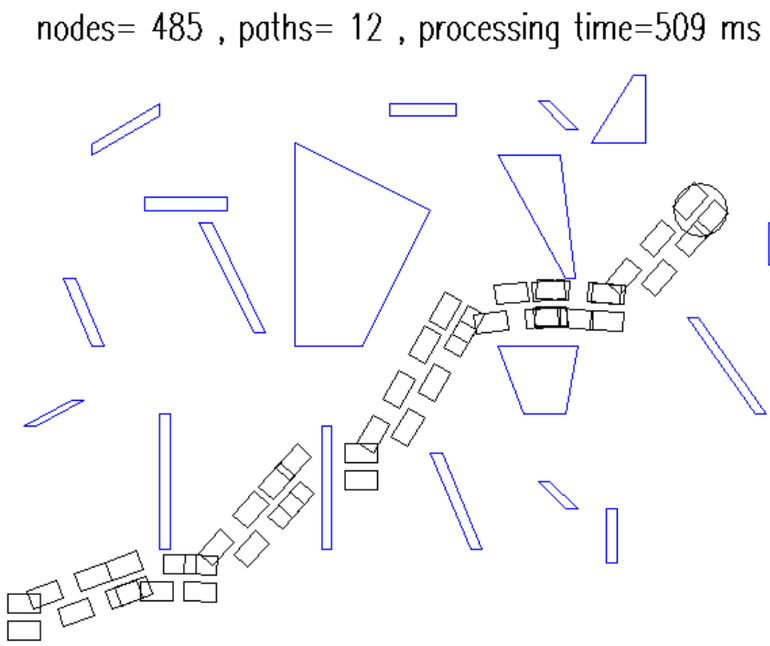


Fig. 13. Best path determined using depth first exhaustive search.

4.8 Search for Best Path

In order to identify the best one of the paths formed by our algorithm, we employ a depth first exhaustive search since greedy or A* search would not filter out for us the best path in every scenario (Cormen, 1994).

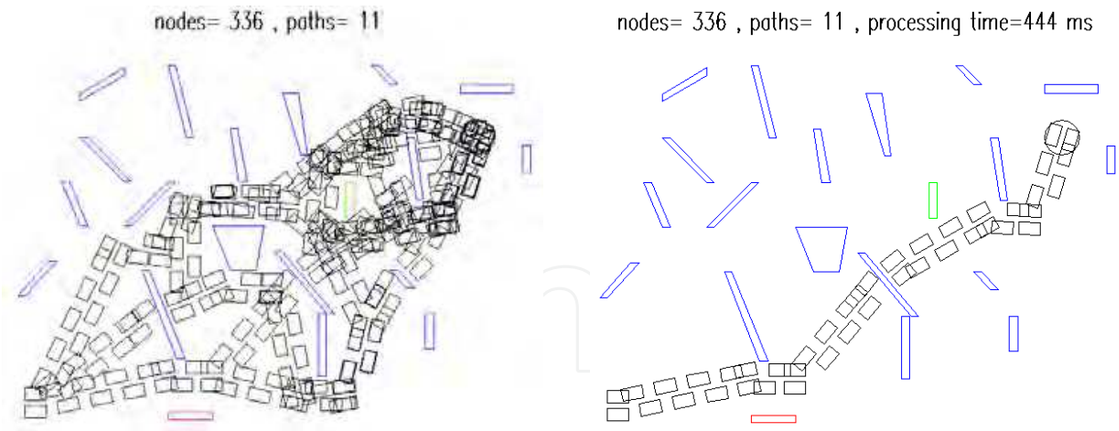
4.9 Simulation Results

Figs. 12 and 13 show results of a simulation of HRP-2 planning footsteps in a room with 20 obstacles. We see that a graph of only 485 nodes is formed consisting of 12 paths all reaching the destination. The whole process takes only 509 ms on a 2.4 GHz Pentium DUO (2 GB RAM) running Windows Vista. A comparison with previous techniques reviewed in section 2 shows reduction in the number of nodes as well as in processing time even though the algorithm employs exhaustive search for hunting out best path which guarantees optimality of the chosen path among those traced out by the algorithm.

Some more simulation results are given in Figs. 14 and 15. We see that the fastest result is obtained in Fig. 14. This corresponds to the lowest number of nodes as well as paths in the graph which also reduces the time taken in search for best path. Fig. 15 shows the trajectories formed around the obstacle of type-3 (drawn with red). It is readily observed that the robot keeps a greater distance with this obstacle than with obstacles of other types.

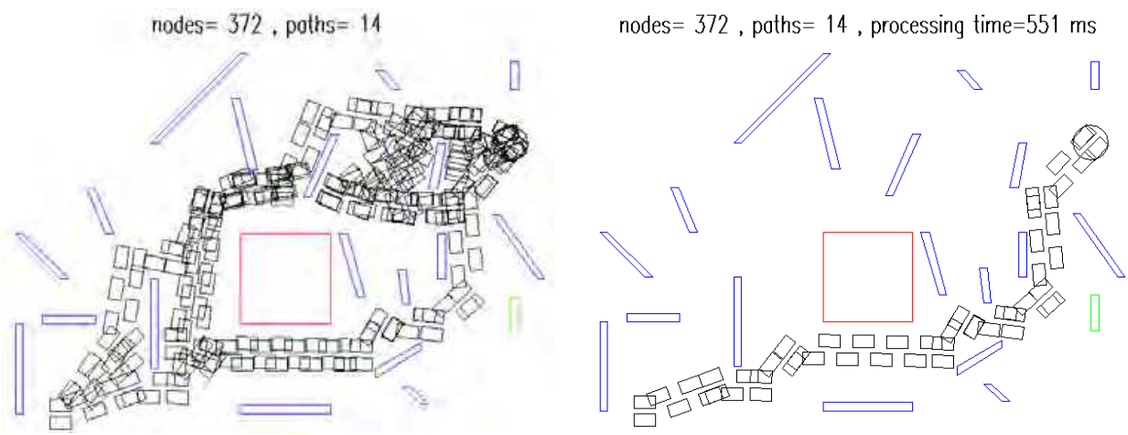
5. Conclusion

Our algorithm successfully demonstrates a novel global reactive footstep planning strategy with a human-like approach. Incremental graph expansion from simpler to more complex



(a) Alternate paths. (b) Best path.

Fig. 14. Result 2: Nodes formed 336, paths 11, time taken 444 ms.



(a) Alternate paths. (b) Best path.

Fig. 15. Result 3: Nodes formed 372, paths 14, time taken 551 ms.

paths ensures formation of a simpler and more useful graph as compared to that formed by approaches such as the visibility graph. The trajectory generated is more energy-efficient since the robot does not have to lift its foot to a high location in every step as in case of game theory based approaches. The algorithm is considerably fast and reduces computational complexity by minimizing the number of alternate steps considered after planning each step. However, basing the cost of each step on energy or time optimization criteria instead of just the complexity level of the stepping motion can further improve the performance of the algorithm. Future work avenues include hardware implementation of the complete footstep planner as well as footstep planning in the presence of moving obstacles.

6. References

Ayaz, Y.; Munawar, K.; Malik, M.B. ; Konno, A. & Uchiyama, M. (2006). Human-Like Approach to Footstep Planning Among Obstacles for Humanoid Robots, Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 5490-5495

Ayaz, Y.; Munawar, K.; Malik, M.B. ; Konno, A. & Uchiyama, M. (2007). Human-Like Approach to Footstep Planning Among Obstacles for Humanoid Robots, International Journal of Humanoid Robotics, World Scientific Publishing Company, Vol. 4, No. 1, pp. 125-149

Ayaz, Y.; Munawar, K.; Malik, M.B. ; Konno, A. & Uchiyama, M. (2007). A Human-Like Approach to Footstep Planning, Humanoid Robots: Human-Like Machines, I-Tech Education & Publishing, pp. 295-314

Chestnutt, J. & Kuffner, J.J. (2004). A Tiered Planning Strategy for Biped Navigation, Proceedings of IEEE Int. Conf. on Humanoid Robotics (ICHR)

Chestnutt, J.; Lau, M.; Cheung, G.; Kuffner, J.J.; Hodgins, J. & Kanade, T. (2005). Footstep Planning for the Honda Asimo Humanoid, Proceedings of IEEE Int. Conf. on Robotics & Automation (ICRA), pp. 629-634

Cormen, T.H.; Leiserson, C.E. & Rivest, R.L. (1994). Introduction to Algorithms, McGraw-Hill Book Co.

Guan, Y.; Yokoi, K.; Neo, E.S. & Tanie, K. (2004). Feasibility of Humanoid Robots Stepping over Obstacles, Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 130-135

Guan, Y.; Neo, E.S.; Yokoi, K. & Tanie, K. (2005). Motion Planning for Humanoid Robots Stepping over Obstacles, Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 364-370

Guan, Y.; Neo, E.S.; Yokoi, K. & Tanie, K. (2006). Stepping Over Obstacles With Humanoid Robots. IEEE Trans. on Robotics, Vol. 22, No. 5, pp. 958-973

Huang, Q.; Yokoi, K.; Kajita, S.; Kaneko, K.; Arai, H.; Koyachi, N. & Tanie, K. (2001). Planning Walking Patterns for a Biped Robot. IEEE Trans. on Robotics, Vol. 17, No. 3, pp. 280-289

Kajita, S.; Kanehiro, F.; Kaneko, K.; Fujiwara, K.; Harada, K.; Yokoi, K. & Hirukawa, H. (2003). Biped Walking Pattern Generation by using Preview Control of Zero-Moment Point, Proceedings of IEEE Int. Conf. on Robotics & Automation (ICRA), pp. 1620-1626

Kajita, S.; Morisawa, M.; Harada, K.; Kaneko, K.; Kanehiro, F.; Fujiwara, K. & Hirukawa, H. (2006). Biped Walking Pattern Generator allowing Auxiliary ZMP Control, Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 2993-2999

Kaneko, K.; Kanehiro, F.; Kajita, S.; Hirukawa, H.; Kawasaki, T.; Hirata, M. ; Akachi, K. & Isozumi, T. (2004). Humanoid Robot HRP-2, Proceedings of IEEE Int. Conf. on Robotics & Automation (ICRA), pp. 1083-1090

Konno, A.; Kato, N.; Shirata, S.; Furuta, T. & Uchiyama, M. (2000). Development of a Light-Weight Biped Humanoid Robot, Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 1565-1570

Kuffner, J.J.; Nishiwaki, K.; Kagami, S.; Inaba, M. & Inoue, H. (2001). Footstep Planning Among Obstacles for Biped Robots, Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 500-505

Kuffner, J.J.; Nishiwaki, K.; Kagami, S.; Inaba, M. & Inoue, H. (2003). Online Footstep Planning for Humanoid Robots, Proceedings of IEEE/RSJ Int. Conf. on Robotics and Automation (ICRA), pp. 932-937

Kuffner, J.J.; Nishiwaki, K.; Kagami, S.; Inaba, M. & Inoue, H. (2005). Motion Planning for Humanoid Robots, Robotics Research, Paolo Dario and Raja Chatila (Eds.), Springer Tracts in Advanced Robotics, Vol.15, pp. 365-374

Latombe, J.C. (1991). Robot Motion Planning, Kluwer Academic Publishers, Boston, Massachusetts, United States of America

McGhee, R.B. & Iswandhi, G.I. (1979). Adaptive Locomotion of a Multi-legged Robot over Rough Terrain, IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-9, No. 4, pp. 176-182

Michel, P.; Chestnutt, J.; Kuffner, J.J. & Kanade, T. (2005). Vision-Guided Humanoid Footstep Planning for Dynamic Environments, Proceedings of IEEE/RAS Int. Conf. on Humanoid Robotics (ICHR), pp. 13-18

Michel, P.; Chestnutt, J.; Kagami, S.; Nishiwaki, K.; Kuffner, J.J. & Kanade, T. (2006). Online Environment Reconstruction for Biped Navigation, Proceedings of IEEE Int. Conf. on Robotics & Automation (ICRA), pp. 3089-3094

Tsujita, T.; Tsuchie, T.; Myojin, T.; Konno, A. & Uchiyama, M. (2007). Development of an Impact Motion Generation Support System, Proceedings of JSME Conf. on Robotics and Mechatronics (RoboMech), ID: 1A1-B01 (in Japanese)

Verrelst, B.; Stasse, O.; Yokoi, K. & Vanderborght, B. (2006). Dynamically Stepping Over Obstacles by the Humanoid Robot HRP-2, Proceedings of IEEE Int. Conf. on Humanoid Robotics (ICHR), pp. 117-123

Vukobratović, M. & Borovac, B. (2004). Zero Moment Point---35 Years of Its Life, International Journal of Humanoid Robotics, World Scientific Publishing Company, Vol. 1, No. 1, pp. 157-173

Yagi, M. & Lumelsky, V. (1999). Biped Robot Locomotion in Scenes with Unknown Obstacles, Proceedings of IEEE Int. Conf. on Robotics & Automation (ICRA), pp. 375-380

Mobile Robots Navigation

Edited by Alejandra Barrera

ISBN 978-953-307-076-6
Hard cover, 666 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

Mobile robots navigation includes different interrelated activities: (i) perception, as obtaining and interpreting sensory information; (ii) exploration, as the strategy that guides the robot to select the next direction to go; (iii) mapping, involving the construction of a spatial representation by using the sensory information perceived; (iv) localization, as the strategy to estimate the robot position within the spatial map; (v) path planning, as the strategy to find a path towards a goal location being optimal or not; and (vi) path execution, where motor actions are determined and adapted to environmental changes. The book addresses those activities by integrating results from the research work of several authors all over the world. Research cases are documented in 32 chapters organized within 7 categories next described.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen