Static Balance for Rescue Robot Navigation: Discretizing Rotational Motion within Random Step Environment

Evgeni Magid and Takashi Tsubouchi

ROBOKEN - Intelligent Robot Laboratory
University of Tsukuba, Japan
{evgeni,tsubo}@roboken.esys.tsukuba.ac.jp

Abstract. The goal of rescue robotics is to extend the capabilities and to increase the safety of human rescuers. During a rescue mission a mobile agent enters a rescue site and it is manipulated by a human operator from a safe place. Without seeing the robot and the environment, a decision on the path selection is very complicated. Our long term research goal is to provide a kind of automatic "pilot system" to propose an operator a good direction to traverse the environment, taking into an account the robot's static and dynamic properties.

To find a good path we need a special path search algorithm on debris and a proper definition of a search tree, which can ensure smooth exploration. In this paper we present our results in estimation of the transition possibilities between two consecutive states, connected with a rotation step. Exhaustive simulations were used to analyze data and to remove unsuitable directions of the search from the search tree. Combining together the results of this paper and our previous results on a translation step and estimation of losing balance on purpose within Random Step Environment, we can now build a search tree and continue toward the path planning process.

1 Introduction

The long standing target of autonomous robotics is to help weak and vulnerable humans dealing with the situations which are beyond our natural abilities: exploring other planets, volcano craters and dangerous old mine tunnels deep under the earth surface, working in a high pressure and poisonous environments of nuclear and chemical plants, substituting human teams in scouting, clear mines and rescue operations. Extending the capabilities and increasing the safety of human teams during search and rescue operations, when victims are often buried in unreachable locations, is the main application of rescue robotics. Instead of sending human rescues to a dangerous site of heavily damaged buildings and underground areas, a rescue robot is deployed there for initial exploration purposes. Then a human tele-operator can operate the robot from a safe place, thus avoiding unnecessary risk of human rescue team casualties from secondary disaster. The system consists of a remote operation station(fig.1(a)) and a mobile robot platform(fig.1(b)), connected with a wireless LAN.

N. Ando et al. (Eds.): SIMPAR 2010, LNAI 6472, pp. 423–435, 2010.

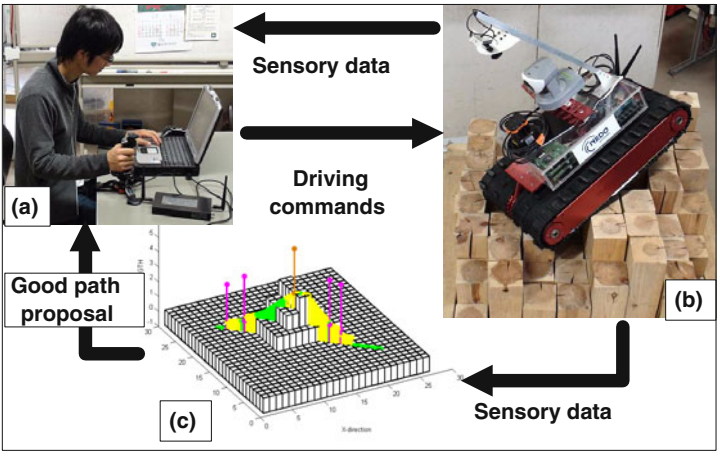

Fig. 1. Standard framework includes operator (a) and rescue robot on RSE (b); we propose to enhance the standard scheme with our "pilot system" (c)

At the moment rescue robots are mainly operated manually by human operators. The operator usually has difficulties to take the decisions based on the limited visual sensory data and choosing a safe robot's path turns into a complicated time consuming process. Staying inside a crawler-type vehicle, the human operator feel the inclination of the slope and the decision on the traversability of the path becomes easier. Unfortunately, the off-site operator cannot use natural biological sensors and judges on the next move only on the base of the available visual information and his/her previous experience. Since many optional paths may connect start and goal locations, it is hard to make an objective decision on a fairly good and safe path. To propose the operator a good path or several options from the current to the goal location is our main research goal. An automatic "pilot system" will calculate the path with regard to the robot's static and dynamic properties. Next, the "pilot system" by means of GUI will suggest several options of a good path to the operator(fig.1(c)). Finally the operator will decide which path to apply it in a real scenario driving the robot.

To find a good path we implement a special path search algorithm on debris. A dangerous and unstable debris site requires the algorithm to keep the robot maximally stable at every step of its path. The real state space of the search is extremely huge. We have to discretize robot's motion and the state space in order to decrease the number of search directions and make the search feasible. A search algorithm utilizes a search tree [2]; for our problem dynamically created search tree can not be explicitly presented as a skeleton. Using as input arguments $Args$ the robot's current configuration and a local environment map, we present the search tree with a function $F(Args) = Res$. Its output Res, a set of accessible within one step configurations, will guide the tree search.

In this paper we present the particular part of function F responsible for a rotational step. We estimate the transition possibilities of a crawler type vehicle

between two consecutive states, connected with a rotation step within Random Step Environment(RSE) - a simulated debris environment, proposed by NIST [4]. Previously we presented results on a translation step between two postures [7] and estimation of losing balance on purpose [8] within RSE. This paper is completing our research of the transition possibilities between two consecutive states and affords us to build a search tree and continue toward a path planning process. Out theoretical results were confirmed with exhaustive simulations, removing all unsuitable directions of the search from the search tree. Currently we are in discussing if the verification of the proposed solution with a large set of experiments within different RSE setups is necessary.

2 Static Balance Estimation of a Posture

Urban search and rescue related research assumes that a rescue robot operates in a dangerous unstable debris site of a heavily damaged by some disaster building. One of the most popular simulations of debris environment is a so-called Random Step Environment(RSE) or Stepfield, proposed by NIST - The National Institute of Standards and Technology [4]. RSE is a set of random steps with equal width and length, but different hight(fig.1(b)). A big variety of optional assemblies, similar to real debris behavior, easy setup and storage made RSE attractive test arenas for evaluation of the urban search and rescue robots performance [9]. RSE is an obligatory test arena for each RoboCup Rescue competition, where at least several RSEs of different difficulty levels are exploited.

Each rescue group uses its own RSE setup, which is more or less similar to an official RoboCup Rescue version. We assemble our RSE from 85mm $\times$ 85mm size wooden blocks of 0, 90, 180 or 270 mm height; 0mm hight corresponds to the ground level around the RSE-patch. We assume a simple tractor-like crawler non-reconfigurable robot with the centroidal location of robot's center of mass (CM). This configuration corresponds to the main body of "KENAF" robot(fig.1(b)), consisting of two large tracks with a small gap in between. Table 1 presents "KENAF" specifications - without sensors, front and back pairs of arms - used in experiments and by the simulation "pilot system".

Table 1. Specifications of "KENAF" in basic configuration

Parameter	Measurement
Maximal inclination	
dynamic	60 deg
static	80 deg
Main body length	584 mm
Main body width	336 mm
Track width	150 mm
Height	270 mm
Weight	17.8 kg

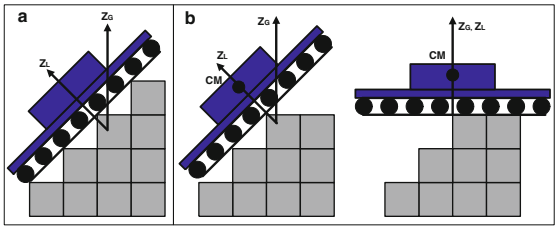

Fig. 2. (a) Green state (b) Orange state: O_1 (left) and O_2 (right)

2.1 Static Stability

The debris site is dangerous and unstable and the main goal of the algorithm is to keep the robot maximally stable at every step of its path in the specific configuration without slippering or turning upside-down. To ensure a stable behavior of the robot while traversing RSE, a continuous satisfaction of static and dynamic constraints[10] is essential. To satisfy a static stability requirement, which is a minimal necessary stability condition[6], the robot's center of mass (CM) must lie above the **support polygon**[1]. For a crawler type vehicle we define a support a polygon as a 2D convex hull of all real contact points of the robot's crawlers with RSE at each given posture.

2.2 Static Balance Estimation

In this section we briefly describe six static balance posture types and assign them color labels, presented in details if [6,7]. **Red state** is statically unstable posture that immediately results in robot's turning upside down if it tries to climb to an impossible steepness. **Magenta state** describes climbing up or sliding down the vertical slope of the environment (fig.3). **Cyan state** is assigned for a robot's jumping down situation during a rotation motion step (if the posture will change withing just one step from fig.3(d) to 3(a)). Statically stable postures(fig.2(a)) are described with a hight quality balance **Green state** and an average quality balance **Yellow state**; Normalized Energy Stability Margin (NESM)[3] is applied to distinguish those two states. To afford the robot losing the balance on purpose we define **Orange state**. This posture does not result in robot's turning upside down, but do not guarantee a single stable posture since two optional postures exist (fig.2(b)). Such behavior is necessary when, for example, the robot traverses a barrier, climbing up and going down with loosing its balance twice on top. Orange state consists of two sub-states : first sub-state O_1 before the robot loses its balance and second O_2, which occurs after the robot loses its initial balance, changes its posture discontinuously at that point and obtains a balance again in a different body orientation. Orange state is a very gentle state and should be used with a special care in translation motion step [7,8] and completely forbidden for a rotational step as we will explain in section 4. Further we denote by R-posture a posture which static balance corresponds to a red state type, M-posture for magenta etc.

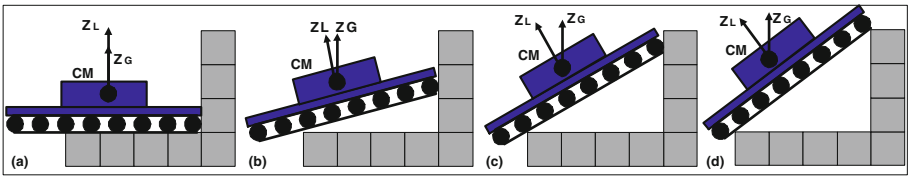

Fig. 3. Magenta state - at climbing up mode robot moves from (a) to (d); at sliding down - from (d) to (a)

2.3 Describing a Posture

To characterize robot's postures qualitatively we use the coloring of the state. To decide on legal transitions between two successive states, we use a combination of 6 following variables[7]. **Steepness** θ_X - the angle, showing the steepness of the RSE at the current robot configuration and rotational moment around robot's transversal Y_L-axis. **Moment** θ_Y - the angle, indicating current rotational moment around robot's lateral X_L-axis. **Contact points quality (CPQ)** depends on the angle θ_{CPQ} between the robot's crawlers and the edges of the RSE cells and affects the robot's ability of climbing the obstacles, losing balance on purpose and going down safely. **Inclination** is the *steepness* angle θ_X sign; with respect to this parameter we specify three groups of posture sets: a climbing up the steps of the RSE posture $G_{U_{inc}}$, a going down posture $G_{D_{inc}}$ and a neutral inclination posture $G_{Z_{inc}}$. **M-sign** is the *moment* angle θ_Y sign; similarly group $G_{P_{MS}}$ is for all postures with positive M-sign, $G_{N_{MS}}$ with negative and $G_{Z_{MS}}$ with neutral postures. Finally, **NESM-stability** - warns about the probability of the robot's turning upside down due to the CM being too close to one of the edges of the support polygon[3].

Inclination and *M-sign* signal about discretization problems, pointing on the missed posture between two successive postures; 4 other variables are emphasized for the experimental work. Combining inclination and M-sign, we specify a neutral Z-posture - a posture with robot's body being parallel to a horizontal patch of RSE : $G_{Z_{inc}} \bigcap G_{Z_{MS}}$. Further we denote each posture as Col(Inc) where Col is the color, Inc is the *inclination*. For example, G(Z) means a green neutral Z-posture and G(U) means G-posture with U_{inc}.

3 Discretizing Search Space and Building a Search Tree

Search tree function $F(Args)$ is to decide on possible next steps of the robot from a given current location and orientation. In a standard 2D navigation each cell of the state space is "free" or "busy" (obstacle, another moving agent etc.) and a transition between two free cells is always legal [5]. In our case we have "possible" (statically stable) and "impossible" (statically unstable) postures. But even in the case of two adjacent "possible" postures the transition between them is not always possible. To decrease the number of search directions we discretize robot's motion and the state space before the search. Starting from 3 levels of

search space discretization for XY-coordinates of the environment, we concluded that discretizing each 85x85mm cell of RSE into 5x5 cells of the internal robot map with the cell size of 17x17mm is the best choice for our problem solution[7].

KENAF supports two types of motion: translation and rotation. Thus at each node of the search tree the search algorithm opens the 3-neighborhood of the node - go straight or turn left/right - and to proceed the search in the most promising direction. All impossible search directions, different for rotation and translation steps, are immediately cut off from the search tree. **Translation step** is defined as a one cell length step forward in the direction of robot local frame's axis X_L. **Rotation step** is a 5 degrees change in robot orientation θ, rotating clockwise(right) or counter clockwise(left). In this paper we present our results in estimation of the transition possibilities between two consecutive states, connected with a rotation step.

4 Rotation Step Transitions

In practice rotational motion within the simulation and in the real world differ a lot. Quality of the surface, number of real contact points, robot's speed, accumulated error in control system, communication delay etc. could result in a high level of imprecision. To ensure that the simulated path could be repeated by the operator in a real scenario, we forbid any dangerous and unpredictable transitions between two states. Appearance of the O-posture is also a hardly predictable situation, so we forbid it for rotational motion and accept losing balance on purpose only at translational step case. To define a legal rotational transition between two stable postures, we created a set of theoretical hypotheses, based on our experimental experience. Exhaustive simulations for environment existence in MATLAB gave a valuable feedback for our theory and generated branch cutting conditions for the path search algorithm(fig.4).

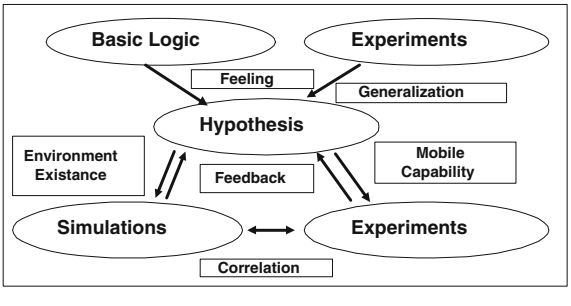

Fig. 4. Theory, simulations and experiments

4.1 Hypotheses

Similarly to a translation step case, described in details in [7], we carried out a set of rotational motion experiments with KENAF robot in several RSEs and

created a set of rules on the rotations (RR) between two successive postures. Some rules are identical for both translation and rotational step. Rules include trivial statements, definitions and assumptions:

(RR1) Neutral posture (Z-posture) is defined if robot's body is parallel to the ground level: $G_{Z_{inc}} \bigcap G_{Z_{MS}}$

(RR2) Perfect rotation preserves a neutral posture and a vertical coordinate Z_{CM} of robot's CM.

(RR3) The only way to climb up or slide down a vertical slope of RSE is to apply a so-called *M-chain* - a sequence of M postures between two stable postures (start G_{start} shown in fig.3(a) and end G_{end} shown in fig.3(d) postures of the *M-chain*). *M-chain* cannot contain any non-M posture.

(RR4) C-posture appears due to discretization problem. With the infinitely small discretization level it would be substituted by a missed *M-chain*.

(RR5) M-posture has no real θ_X, θ_Y and NESM parameter data, but only an approximation. Also it has no own inclination and M-sign data.

(RR6) Loosing balance on purpose through O-posture during rotational step results into unpredictable next posture of the robot. In a worst case it results into robot's turning upside down.

(RR7) Change of inclination between two postures can occur only through O-posture and its O_1 or O_2 is a Z-posture.

(RR8) Change of M-sign can occur only through O-posture.

(RR9) Any significant change in 3D orientation of the robot may result into robot's turning upside down.[1]

4.2 Successive Posture Groups

We divided all possible pairs of postures, connected with a rotational step, into groups. Each group contains theoretically possible or impossible sequence; some of possible sequences are strongly recommended, while some are undesirable. For the rotation case a legal set of colors for each posture is $\{G, Y, O, M, C\}$ and we are supposed to obtain 25 groups of pairs at most. To decrease the number of groups we treat at the simulation level G and Y as a same G color.

Forbidden Sequences

In addition to appearances of R-postures, any pair, containing O-posture, is forbidden with regard to (RR6). If during a 5-degree rotation step robot will lose its balance at O_1 in order to obtain it again at O_2, real robot's behavior on RSE becomes unpredictable and loosing balance on purpose may result in turning upside down. This implies not only an explicit appearance of O-posture, but also a missed O-postures mentioned in (RR7) and (RR8). Also we forbid any rotational step which results in robot's climbing or jumping up situation. Fig.5 demonstrates the example of jumping up between two successive postures, connected with a 5 degree rotational step: current posture(left) and next posture(right). In most cases it is enough to check the vertical change in CM-coordinate Z_{CM} to

[1] We forbid any change exceeding 4 degrees. For the details see section 4.2.

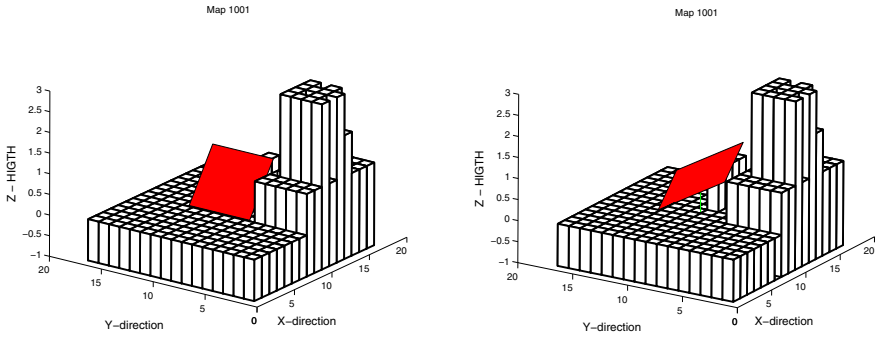

Fig. 5. Jump up case: current(left) and next(right) postures

detect a jump up: any case exceeding $\varepsilon \doteq 4.5$mm used to take into an account accumulated numerical error, is physically impossible. No jump up or down between two neutral postures G(Z) → G(Z) and no climbing up between G_{start} and G_{end} ends of M-chain can occur. Any jumping down, except C-case, is forbidden as well. C-case permits a jump down within physically acceptable vertical jump of the robot. There is also a number of other postures, which we define as forbidden postures and they will appear later in this section.

Undesirable Sequences
Similar to O-posture case, M-chains and single M-postures are not predictable enough even for the case of sliding down a vertical slope of RSE. Main difficulty is that since we cannot detect precisely where in [0,5] degrees rotational interval of the 5-degrees rotation the robot started to slide down (a start of M-chain), we cannot predict the result of the process. The same explanation, with regard to (RR4), makes any sequence, containing at C-postures, undesirable. Being physically possible, these two postures do not result in robot's turning upside down and do not destroy robot's sensors, the most vulnerable part of the robot, while sliding down. Yet we would like to abstain from their appearance in the path because we can not predict exactly what will be the next posture of the robot. The only case when we permit using undesirable sequences during the path search process is when no other better choice is available. However, if choosing an undesirable sequence, upon arrival to a newly obtained after sliding down posture, we will have to recollect the sensory data about the local RSE patch, localize the robot and restart the path search process starting this new posture.

Perfect Sequence
We distinguish 3 types of possible sequence: perfect, good and fair. Perfect case is a rotation on a flat pattern of RSE with current and next both neutral postures and constant location of CM in vertical direction Z_{CM}, described by (RR2).

Good and Fair Sequences
All other G → G cases form good and fair cases group or are added to undesirable and impossible cases groups. To make this more precise division of the cases we apply parameters, defined in section 2.3. For all non perfect G → G transitions

Table 2. G → G posture types distribution

GG1	GG2	GG3	GG4	GG5	GG6	GG7	GG8
45%	0.02%	0.02%	0.21%	0.01%	0.15%	2.7%	51.87%

we apply a special contact points checking procedure, which compares expected contact points vertical locations between current and next postures. We forbid the transition if at least one of the expected contact points jumped up.

Combining different inclinations we obtained 8 groups of G → G postures:

(GG1) G(D) → G(D) (GG2) G(D) → G(Z) (GG3) G(D) → G(U)
(GG4) G(Z) → G(D) (GG5) G(Z) → G(U)
(GG6) G(U) → G(D) (GG7) G(U) → G(Z) (GG8) G(U) → G(U)

We analized the statistics[2] for the G → G postures groups appearance, presented in table 2. Subgroups GG3 and GG6 are forbidden with regard to (RR6) and (RR7). GG1 and GG8 cases, being a majority of the groups (45% and 51.87% respectively), were expanded further. Suspicious groups, forming all together 2.94% of the G → G cases, were excluded from further expansion and added to forbidden cases. First of all, if we want to use GG1 and GG8 cases for the path planning, we must remove any dangerous possibilities. Forbidding jump up of ε = 4.5mm is not enough; to increase the level of safety, we forbid *any* jump up exceeding zero. Of course, we aware that in many cases we forbid possible cases obtained due to a accumulated numerical error, but we can not take a chance to accept a real jump up. Four no jump up (ok) and jump up (j) sequences have almost the same appearance about 25% of the cases(table 3). Again, we point at the important issue: we take no risk of getting stuck in the real world, accepting suspicious cases in the pilot system at the simulation level.

Table 3. GG1 and GG8 : no jump up (ok) and jump up (j) sequences distribution

GG1(ok)	GG1(j)	GG8(ok)	GG8(j)
25.44%	21.01%	28.37%	25.18%

We are interested to apply the rotations which preserve robot's body 3D orientation and avoid the cases of extreme orientation changes, which in the worst case result into robot's turning upside down. As a measure of this change we use a support polygon's plane normal, which coincides with a Z-axis of a local coordinate frame of the robot Z_L. The change in normal orientation between two successive postures as a measure of orientation change:

$$\Omega = |\widehat{Z_G Z_L}(curr) - \widehat{Z_G Z_L}(next)| \quad (1)$$

where $\widehat{Z_G Z_L}$ is an angle between the Z-axis of the global RSE frame Z_G and Z_L(curr) and Z_L(next) correspond to the Z-axis of a local frame at current and

[2] Simulation setup for statistics is described in details in section 5.

next postures respectively. We distinguish 5 distinct subgroups for Ω parameter for each of 2 cases GG1(ok) and GG8(ok):

(a) $\Omega \in$ [0,1)
(b) $\Omega \in$ [1,2)
(c) $\Omega \in$ [2,3)
(d) $\Omega \in$ [3,4)
(e) $\Omega \in$ [4,90)

Voting for those 10 subgroups of GG1(ok) and GG8(ok) sequences showed the following distribution (table 4). Based on the statistics we divide GG1(ok) and GG8(ok) sequences into 3 main groups:

- GG1(a) and GG8(a) form good sequences - they appear in 86.01% cases.
- GG1(b,c,d) and GG8(b,c,d) form fair sequences, where the quality decreases as Ω increases; all together 13.4%.
- GG1(e) and GG8(e) are forbidden sequences with $\Omega > 4$ degrees 0.58%. Even though some of them may be still possible, their removal from the path search procedure will not seriously influence the quality of the path as statistics shows. Currently we are discussing if we need to increase the lower bound of the interval through verification experiments.

Table 4. GG1(ok) and GG8(ok) sequences percentage with regard to parameter Ω

GG1(a)	GG1(b)	GG1(c)	GG1(d)	GG1(e)	GG8(a)	GG8(b)	GG8(c)	GG8(d)	GG8(e)
40.64	4.01	1.21	1.17	0.26	45.38	4.8	1.09	1.13	0.31

4.3 Summary

Summarizing, we group all rotational step cases into 5 clusters:

1. Forbidden - R and O-postures, all jumping and climbing up cases, groups GG2 GG7, suspicious to a jump up subgroups GG1(j) and GG8(j) and big change in 3D orientation subgroups GG1(e) and GG8(e)
2. Undesirable - Sliding down M-chains and jumping down C-postures; they could be included as a part of the path if no better option exists.
3. Perfect - G(Z) $\rightarrow$ G(Z) transition with no vertical change in CM location
4. Good - subgroups GG1(a) and GG8(a), where the change in 3D orientation does not exceed 1 degree
5. Fair - subgroups GG1(b,c,d) and GG8(b,c,d), where the change in 3D orientation is between 1 and 4 degrees

5 Simulations

Only an experimental proof can guarantee that a good theoretical hypothesis will work correctly in a real world as well. A huge number of various cases can happen in a completely random RSE and implementation of such amount of experiments is physically impossible. Impossible in the real world pairs of postures are also

impossible within the simulation. As far as the reverse statement is not true, the simulator cannot replace the experiments, but assists to structure the data and remove the impossible types of sequences, saving time and efforts. Successive transition patterns will be integrated in the search algorithm as a part of neighbor opening and branch cutting function $F(Args) = Res$.

For the exhaustive check we created a huge 61x61 cell map (fig.6) which includes all typical obstacles, usually appearing in the RSE: horizontal and diagonal barriers, pairs of parallel barriers, valleys, traversable and non-traversable pikes and holes. All possible pairs of postures connected with a rotation step were proceeded with voting for each group, described in section 4. For a first posture of the pair we placed robot's CM at every node of the grid; next posture was calculated as a 5 degree increment of the robot's heading direction θ, rotating right. The simulation included 86 initial robot orientations $\theta \in \{0, \frac{\pi}{180}, \frac{2\pi}{180}, ...\frac{84\pi}{180}, \frac{85\pi}{180}\}$. We assume a symmetrical behavior for rotating left/right and for other 3 quarters of the 360-degree rotation range. In addition to pointing at the impossible (empty) cases the simulation reveals the rare cases and the most often cases.

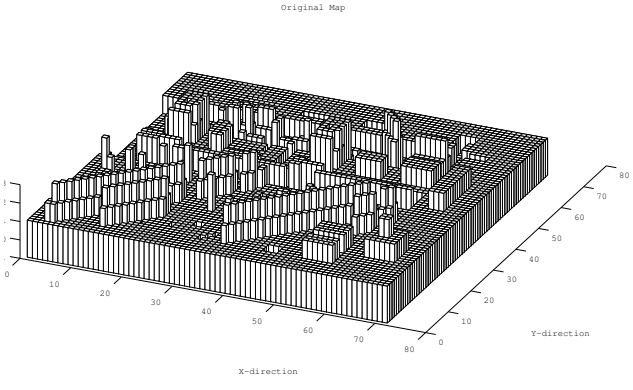

Fig. 6. A 61x61 cells RSE, covering all main types of the environment obstacles

The total number of posture pairs was more then 6 millions. The results of the simulation are summarized in table 5. Among them O-posture appeared in 1.58% of the pairs explicitly and in 0.06% as a missed O-posture. Impossible pairs with a significant jumping up/down behavior form together 1.12% cases. Excluded pairs, which were later also added to forbidden cases, are 0.94%. Together with all additional forbidden posture sequences, forbidden group reaches 23.59%. Undesirable M and C-postures with sliding down behavior are only 2.36%. Perfect rotations are the most significant amount of the cases, 56.64%; good rotations score 15.71%, while fair rotations sum up to 0.76% only. This statistics matches well with our expectations based on the operational experience - rotational step is also a hard challenge for the operator and in the simulation only 73.11% of the pairs were detected to be more or less suitable as the path points; 2.36% of the pairs are hardly suitable and the rest 24.53% are completely useless.

Table 5. Main, undesirable and forbidden transitions

Perfect	Good	Fair	Undesirable	Forbidden	Excluded
56.64	15.71	0.76	2.36	23.59	0.94

6 Conclusions and Future Work

To provide an assistant "pilot system" for an operator of a rescue robot, which will be able to propose the operator a good path from the current to the target location, is our main long term research goal. After obtaining data from the environment a robot creates an internal world model and the path selection within the internal model should be done. Since usually there exist more then just a single path, a good instrument to evaluate the quality of each path is important for the path search algorithm.

The search algorithm within the graph requires a proper definition of neighboring states to ensure smooth exploration of the search tree. In this paper we presented our results in estimation of the transition possibilities between two consecutive states, connected with a rotation step. It is our final step toward a definition of a search tree neighborhood function $F(Args) = Res$, where arguments $Args$ are the robot's current configuration and the environment and output Res is a set of accessible within one step configurations. Our theoretical basis for function F was confirmed with exhaustive simulations, removing unsuitable search directions. Next we consider to verify our results with experiments. Combining together the results of this paper and our previous results on a translation step and estimation of losing balance on purpose within RSE, we can now build a search tree and continue toward the path planning process.

Acknowledgments

This research has been partially supported by NEDO Project for Strategic Development of Advanced Robotics Elemental Technologies, High-Speed Search Robot System in Confined Space.

References

1. Bret, T., Lall, S.: A Fast and Adaptive Test of Static Equilibrium for Legged Robots. In: IEEE ICRA 2006, pp. 1109–1116 (2006)
2. Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to Algorithms, 2nd edn. The MIT Press/McGraw-Hil (2001)
3. Hirose, S., Tsukagoshi, H., Yoneda, K.: Normalized energy stability margin: generalized stability criterion for walking vehicles. In: Proc. of 1st Int.Conf. On Climbing and Walking Robots, Brussels, pp. 71–76 (1998)
4. Jacoff, A., Messina, E., Evans, J.: Experiences in Deploying Test Arenas for Autonomous Mobile Robots. In: Proc. of the 2001 PerMIS Workshop, in association with IEEE CCA and ISIC, Mexico City, Mexico (2001)

5. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers, USA (1991)
6. Magid, E., Ozawa, K., Tsubouchi, T., Koyanagi, E., Yoshida, T.: Rescue Robot Navigation: Static Stability Estimation in Random Step Environment. In: Carpin, S., Noda, I., Pagello, E., Reggiani, M., von Stryk, O. (eds.) SIMPAR 2008. LNCS (LNAI), vol. SIMPAR, pp. 305–316. Springer, Heidelberg (2008)
7. Magid, E., Tsubouchi, T.: Static Balance for Rescue Robot Navigation: Translation Motion Discretization Issue within Random Step Environment. In: Proc. of ICINCO, Madeira, Portugal, pp. 305–316 (2010)
8. Magid, E., Tsubouchi, T., Koyanagi, E., Yoshida, T.: Static Balance for Rescue Robot Navigation: Losing Balance on Purpose within Random Step Environment. In: Proc. of IEEE IROS, Taipei, Taiwan (accepted, 2010)
9. Sheh, R., Kadous, M., Sammut, C., Hengst, B.: Extracting Terrain Features from Range Images for Autonomous Random Stepfield Traversal. In: IEEE Int. Workshop on Safety, Security and Rescue Robotics, Rome, pp. 1–6 (September 2007)
10. Shoval, S.: Stability of a Multi Tracked Robot Traveling Over Steep Slopes. In: IEEE ICRA 2004, vol. 5, pp. 4701–4706 (2004)