Proceeding of the IEEE
International Conference on Robotics and Biomimetics (ROBIO)
Shenzhen, China, December 2013

Simulated Capacitor Method for Difficult Region Dynamic Boosting in Changing Environments

Hong Liu, Tianwei Zhang, Yan Wang and Zhaopeng Gu*

Abstract— **This paper presents a novel method to identify and boost difficult regions of in the configuration space (C-space) in changing environments. Difficult regions, especially narrow passages, change their shapes frequently in changing environments, which result in challenging problems to find valid and safe paths. Although a lot of research has been done to identify difficult regions, seldom methods provide robust paths. Moreover, they depend on frequent replanning which wastes a lot of resources. In this paper, a bridge test method based on workspace to configuration space (W-C) nodes mapping is invoked to identify difficult regions dynamically. Consequently, not only difficult regions are identified efficiently, but safe regions which are less likely to be occupied temporarily, are flagged for boosting. Furthermore, the calculated simulated capacitance, which represents the local difficulty level is used to lead boosting procedure. Specifically, boosting nodes would be activated or re-closed according to the changing capacitance, so that calculation resources are concentrate on current difficult regions. As a result, both the total planning time and replanning times are descended a lot. Finally, the simulation experiment with two manipulators shows the proposed method is efficient even in difficult changing environments.**

I. INTRODUCTION

Sampling-based methods, such as Probabilistic Roadmap Method (PRM)[1] and Rapidly exploring Random Trees (RRT)[2], contributes a lot in robotics path planning. However, sampling-based methods show their weakness in some special regions, such as narrow passages and obstacles' boundaries, due to their small volumes. In recent twenty years, plenty of variants of PRM and RRT [3], [4], [5], [6] have been proposed to deal with these difficult region problems and have achieved great success even in high-dimension configuration space (C-Space). Nevertheless, there are still challenging problems in changing environments, because difficult regions change their shapes when obstacles move.

Hong Liu is with the Engineering Lab on Intelligent Perception for Internet of Things (ELIP), Shenzhen Graduate School, Peking University, China hongliu@pku.edu.cn

Tianwei Zhang is with the Engineering Lab on Intelligent Perception for Internet of Things (ELIP), Shenzhen Graduate School, Peking University, China zhangtianwei@sz.pku.edu.cn

Yan Wang is with the Engineering Lab on Intelligent Perception for Internet of Things (ELIP), Shenzhen Graduate School, Peking University, China yanwang@sz.pku.edu.cn

Zhaopeng Gu is with the Engineering Lab on Intelligent Perception for Internet of Things (ELIP), Shenzhen Graduate School, Peking University, China guzp@pkusz.edu.cn

This work is supported by National Natural Science Foundation of China (NSFC, No.60875050, 60675025), National High Technology Research and Development Program of China (863 Program, No.2006AA04Z247), Scientific and Technical Innovation Commission of Shenzhen Municipality (No.JCYJ20120614152234873, CXC201104210010A, JCYJ20130331144631730, No.JCYJ20130331144716089).

Dynamic Roadmap Method (DRM)[7] tailors the PRM framework to make it adapt to changes occurring to the roadmap. It re-validates points and edges by a precomputed mapping from W-space to C-space. To deal with changing difficult regions, DRM-based methods require more incremental sampling points inside. We conclude two crucial issues for dealing with this problem: (1) How to identify difficult regions instantly. (2) How to effectively increase density of C-free points in difficult regions. Many studies have focused on the above two points [10], [19].

Although many works have achieved significant development in dealing with difficult regions, most of them neglect path safety, which is also very important for realtime planning [12], [13]. Specifically, method of [17] puts calculation resource into localized subgoals to improve single planning efficiency, and when the environment changes, replanning is invoked to find a new path. Without considering paths safety, the selected paths are usually invalidated by changing obstacles, resulting in frequent replanning and poor global planning efficiency.

In this paper, a novel Simulated Capacitor Method is proposed to identify and boost difficult regions in changing environments. This method contains two main steps: Simulated Capacitor Formation and Capacitance Boosting. When obstacles move in W-space, capacitor bridges are built between the positive and negative toggled points to locate narrow passages and obstacle boundaries. Then, simulated capacitors are formed and capacitances are calculated to represent the difficulty level. Moreover, for increasing density of C-free points in difficult regions, boosting samples which are preprocessed would be activated partly according to the capacitance. As capacitances are updated in time, boosting points around less difficult regions will be shut down, so that the planner maintains an acceptable point number, avoiding a waste of calculation resource and improving realtime planning efficiency. In addition, boosting points are generated around positive toggled points to enhance path safety. Therefore, safe paths with less probability to be occupied by obstacles will be found out to avoid extra replanning.

Main contributions of this paper can be concluded as follows:

(1) Simulated Capacitor Method is proposed to instantly identify difficult regions in changing environments, difficult regions are flagged out and capacitances are calculated to indicate the difficulty level.

(2) Capacitance Boosting Strategy is proposed to effectively increase density of C-free points in difficult regions,

978-1-4799-2744-9/13/$31.00 ©2013 IEEE

providing safe paths to avoid extra replanning and regulating the validity of boosting points to enhance planning efficiency.

The rest of this paper is organized as follows: Section II shows related works. Details of our method are described in Section III and Section IV. Experiments are drawn in Section V, and conclusions are given in Section VI.

II. RELATED WORK

A. DRM

DRM is a modified variant of PRM to adapt to changing environments. It generates nodes randomly since there is no obstacle considered initially. The gist of DRM is to represent the relationship between W-space and a roadmap in C-space by means of constructing two kinds of mapping, nodes mapping (1) and edges mapping (2):

$$\Phi_n(w) = \{q \in G_n \mid \Omega(q) \cap w \neq \emptyset\} \tag{1}$$

$$\Phi_a(w) = \{\gamma \in G_a \mid \Omega(q) \cap w \neq \emptyset \ for\ some\ q \in \gamma\} \tag{2}$$

here, $G = (G_n, G_a)$ is the roadmap constructed in C-space. G_n is the set of nodes and G_a is the set of edges. $\Phi_n(w)$ and $\Phi_a(w)$ indicate the nodes and edges of the roadmap which are invalid caused by the basic cell w of W-space occupied by obstacles, respectively. $\Omega(q)$ denotes a subset of basic cells occupied by robot whose configuration is q.

In contrast to computing the complex mapping $\Phi_n(w)$ and $\Phi_a(w)$, the inverse mapping Φ_n^{-1} and Φ_a^{-1} are computed. For example, to compute Φ_n^{-1}, the robot in the W-space is first set to the configuration in C-space, and then a seed cell is put inside the robot and expanded in each direction until all cells $\Omega(q)$ occupied by the robot are found by collision checks. The computing of Φ_a^{-1} is to make edge γ discrete recursively until a required resolution is reached. Generally speaking, computing edges mapping costs too much and is less important compared with W-C nodes mapping.

In spite of the significant development of DRM in path planning for changing environments, the probability of finding free path is low in the case of existing narrow passage in C-space, due to the fact that DRM has sampling bias in difficult region initially.

B. Safe Planning

Safe motion planning is of importance in improving robots' safety and reducing planning cost. Finding a safe path is more reliable than obstacle avoidance and replanning manners. In [11], computation of the Regions of Inevitable Collision (RIC) was proposed to find a safe path by predicting whether a region would be occupied by obstacles or not. The notion of Near and Potential Collisions Regions is also introduced, which represented potentially dangerous states that are heuristically evaluated according to planning risk. This method shows superiority in low-dimension planning. However, due to the complexities of approximate computation and discretization, it is too difficult to apply this method in high dimensional problems. [18] planned out of RIC by selecting a proper time horizon for the velocity obstacle. This time horizon was determined by the minimum time the robot needed to avoid collision, either by stopping or by passing the respective obstacle. In spite of these, their planner was sensitive to obstacle's shape, velocity, and path curvature, which are difficult to deal with in 3D scenarios.

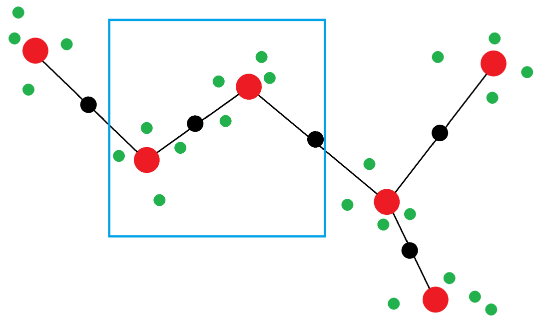

Fig. 1. Hierarchical Sampling Points

Though several works focusing on safe planning, no specific or uniform criteria of path safety have been proposed. The safe path in this paper, means not only collision-free, but also less probable to be occupied by obstacles before reaching the goal.

C. Hierarchical Sampling Strategy

Based on W-C nodes mapping, a hierarchical sampling strategy is applied to describe the structure of C-space and increase the nodes density in difficult regions which are located. To save the calculating resource, nodes will be pre-generated in different levels. The first and second level points P and M are live points, which are sampled to describe the structure of C-space. P would be used as flags in each updating phase, and M are middle points of P. In addition, the last level incremental points B are generated as dead points, which would be activated after appearance of flags during the updating phase. Although only a part of B are maintained, the quantity of B will ascend with the robot moving. Therefore, the planner becomes slower and slower when the robot moves to unfamiliar places.

1) Sampling Progress: The hierarchical sampling progress is shown in Fig.1. At first place, point set P , colored in red in Fig.1, are generated by uniform random sampling without obstacles in C-space. Let $P = \{p_1, ..., p_n\}$, represents all the first level points. For each node $p \in P$, connect K edges E_n with its K nearest neighbor nodes, the number K is set in advance. At second place, for each edge $e \in E_n$, compute the middle node $m \in e$, which is colored in black, from its endpoints' coordinates. Let $M_n = \{m_1, ..., m_i\}$, represents middle points of p_n. Number i depends on the amount of edges of p_n. Finally, the last level green points are generated around P. For each $p \in P$, K points b_n are generated by random sampling within the radius R. Distance R is the average distance between p and its middle points, which makes b well-distributed. Let $B_n = \{b_1, ..., b_k\}$ represents boosted points belonging to

p_n. Here, B_n contains structural information of difficult regions if p_n is bridged. For each $b \in B$, connect K edges with its K nearest P and M points. What's more, all points and edges generated in Step-3 will be set inactive. They will be activated during the updating phase if their main points $p_n \in P$ are selected. Distance metric has a significant influence on sampling based methods [15], [16]. Hence, all distances mentioned in this paper are weighted Manhattan distances, which properly reflect the distance information between each two C-space nodes.

2) Incremental Points Updating: The validity of main points P and M is updated by W-C nodes mapping. As the incremental points B are not mapped and online collision checking is consuming, we use a predictive model, called Inner Parzen Window [9], to predict the validity of a node according to its mapped neighbors. In the updating phase, for each node activated b, this algorithm will compute its probability of validity $P(b)$ in a Parzen Window ($IPWindow$) centered at b. $P(b)$ is defined as:

$$P(b) = \frac{\sum^{window} N_{valid}(P+M)}{\sum^{window} N(P+M)} \tag{3}$$

here, $N_{valid}(P+M)$ represents the number of valid nodes belonging to set P and M in Inner Parzen Window ($IPWindow$) area. $N(P+M)$ is the total number of nodes belonging to set P and M in the $IPWindow$ area. r_{window} is the radius of $IPWindow$ and is set to be $2R$, which has been discussed in Part A, Section III, to ensure that $IPWindow$ at least encloses one sample point. Incremental point b will not be really added to the roadmap unless $P(w) > Threshold$, and the $IPWindow$ can be substituted by the nearest K points of P and M.

III. Simulated Capacitor Algorithm

To find difficult regions in changing environment, CBB method is proposed in [20]. It is based on a W-C nodes mapping in the preprocessing phase, and updates each node's validity without collision checking in the updating phase. In the query phase, an A* method is employed to find the optimal path in the roadmap.

CBB provides both efficiency and accuracy for difficult regions identification, but it doesn't provide the difficulty level information, which is a significant index to conduct further boosting. If we boost all difficult regions with the same level, a lot of calculation resource would be wasted.

A. Capacitor Bridge

In the preprocessing phase, three-level nodes are sampled and then W-C mapping is built with the main nodes. Specifically, there are two steps to build this mapping: (1) Decompose W-space into small cells. (2) Compute $\Phi_n^{-1}(q)$. Computation of $\Phi_n^{-1}(q)$ has been described in part A of Section II.

In the updating phase, validity of each node around the obstacle may toggles when the obstacle moves. These transformation information can be immediately acquired with

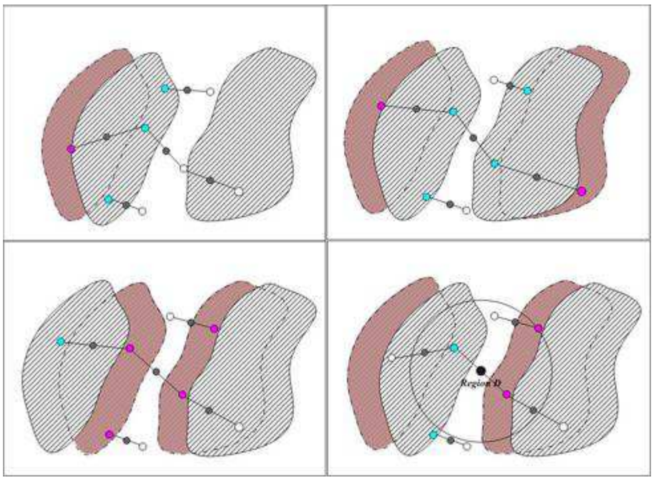

Fig. 2. Simulated Capacitor

W-C nodes mapping. We define the point set $P+$, as points which have validity toggle from false to true, and $P-$, as points which have validity toggle from true to false. When obstacles move in W-space, sample points which have been released by them have positive toggles: validity changes from false to true. While sample points which have been just occupied by them have negative toggles: validity change from true to false.

If one edge e has both $P+$ and $P-$ endpoints, and $m \in e$ is valid, a simulated capacitor is formed to flag a narrow passage. Fig.2 shows the narrow passage identify principle of capacitor method. Specifically, when an obstacle moves to a new position, as well as its configuration, the red shadow region represents its previous position. By the help of W-C nodes mapping, modification of P can be obtained. For each $p \in P$, if $p.validity$ toggles from false to true, add p to $P+$ and colore it in purple. Otherwise, if $p.validity$ toggles from true to false, it will be put into $P-$ and colored in blue. There are four situations of a C-space narrow passage in Fig.2. For the upper two pictures, one obstacle moves to another, and there are only blue $P-$ near the narrow passage, so this narrow passage will not be flagged due to the fact that two islands may bump together and this C-free region is dangerous. The third situation in which two obstacles move away from each other will not be flagged, ether. Because there has been several available nodes and there will be more and more later, boosting is not necessary. Then, the last situation in which the narrow passage is changing with the moving obstacles should be flagged. The detail method is as follows: for each $p \in P+$, its middle point $m \in M_n = \{m_1, ..., m_k\}$ are found by edges of $E_n = \{e_1, ..., e_k\}$. If $m.validity$ is true, find the other endpoint p' of e. If $p' \in P-$, a "capacitor" bridge will be built, and p will be marked as a flag to indicate that Region D is a narrow passage. Details are shown in Algorithm 1.

Algorithm 1 Simulated Capacitor Builder

Require: W-C nodes mapping for P and M

Updating phase:
for each node $p \in P$ **do**
  if $p.validity$ turns from false to true **then**
    $p \in P+$
  else if $p.validity$ turns from true to false **then**
    $p \in P-$
  end if
end for
for each node $p \in P+$ **do**
  Pick each edge e connected with p, get middle point $m \in e$
  if $m.validity$ is true **then**
    Get the other endpoint $p' \in e$
    if $p' \in P-$ **then**
      mark (p, p')
    end if
  end if
end for

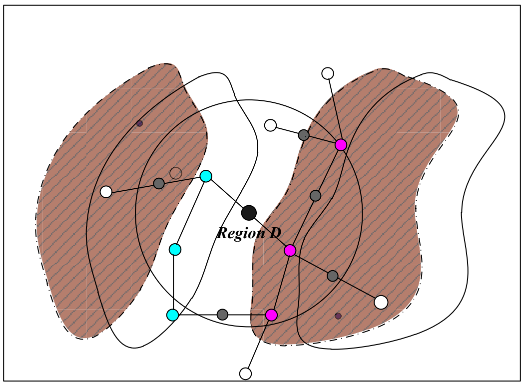

Fig. 3. Simulated Capacitor

B. Capacitance Calculating

As we know a simple capacitance can be calculated by:

$$C = \frac{\varepsilon \cdot S}{4\pi k d} \quad (4)$$

In formula (4), S can be understood as charge density, here we use the sum number of $P+$ and $P-$ to describe it; the distance d in a simulated capacitor is the weighted Manhattan distance, which has been calculated in preprocessing phase; the constants are used to normalization. Therefore, the simulated capacitance can be calculated as follows:

$$C = \varepsilon \cdot \frac{\sum(P_s+) + \sum(P_s-)}{[\alpha \cdot \sum_{i=1}^{n} |d_i - d_i'|^2]^{\frac{1}{2}}} \quad (5)$$

For instance, in Fig.3, there are two capacitor bridges between the narrow passage, assume the capacitance of the central one is C_1, the bottom one is C_2. As discussed above, side length d_1 and d_2 have been calculated in the preprocessing phase. Here, assume d_1 = d_2 = d. Then, there are two $P+$ points connected with C_1's positive endpoint and one $P-$ point connected with its negative endpoint. As a result, C_1 and C_2 can be expressed as $(2+1)\varepsilon/d = 3\varepsilon/d$ and $(1+1)\varepsilon/d = 2\varepsilon/d$, respectively. Clearly, the central place is harder than the bottom place in this narrow passage, and these changing capacitance will be used to adjust the coming boosting. The algorithm is shown in Algorithm 2.

Algorithm 2 Capacitance Calculation

Require: (p, p') got in Algorithm 1

Preprocessing phase:
for each $e \in E$ **do**
  $d = [\alpha \cdot \sum_{i=1}^{n} |d_i - d_i'|^2]^{\frac{1}{2}}$
end for
Updating phase:
for each (p, p') **do**
  for each node p'' connected with p **do**
    if $p'' \in P+$ **then**
      $\Sigma(P_s+)++$
    end if
  end for
  for each node p''' connected with p' **do**
    if $p''' \in P-$ **then**
      $\Sigma(P_s-)++$
    end if
  end for
end for
for each node $F(p, p') \in F$ **do**
  calculate the capacitance with $c_n = \varepsilon \cdot \frac{\sum(P_s+)+\sum(P_s-)}{[\alpha \cdot \sum_{i=1}^{n} |d_i - d_i'|^2]^{\frac{1}{2}}}$
  $c_n \in C\{C_1(p_1, c_1), \ldots C_m(p_m, c_m)\}$
end for

IV. Capacitance Boosting Strategy

As difficult regions are flagged by capacitor bridges, the next task is to improve node density inside them. Positive half bridge regions near $P+$ are less likely occupied by obstacles, therefore, Capacitance Boosting Strategy is proposed to conduct the boosting process. Incremental points of B, which have been pre-sampled, would be activated according to the capacitance, and the validity of B would be obtained by a predictive model, which has been discussed in Section II.

For each flag $p \in P+$ obtained in updating phase, only one safe endpoint of the bridge $p \in P+$ is boosted. Since P are generated uniformly, $P+$ and $P-$ will be regenerated as long as obstacles move. $P+$ always follow obstacles, while $P-$ always stand in the way of obstacles. Therefore, simulated capacitors formed in each updating phase always follow a wall of the narrow passage and avoid another one. If two walls move toward each other, no bridges will be built there, thus indicating the corresponding narrow passage will disappear soon.

For example in Fig. 4, a narrow passage is marked by capacitor bridge. The red region is the previous position of the C-obstacle, and the gray is the new position of it. Both left and right side have several main points, and white points

Algorithm 3 Capacitance Boosting Strategy

Require: $C\{C_1(p_1,c_1),\ldots C_m(p_m,c_m)\}$

Updating phase:
for each node $b \in$ UpdateArray **do**
  $b.validity$ = false
  for each $e \in b$ **do**
    $e.validity$ = false
  end for
end for
clear UpdateArray
for each node $C_n \in C$ **do**
  $m = 5 \cdot c_n$ Pick $B_n \in p_n$
  for each $b \in B'_n\{b_1,\ldots,b_m\}$ **do**
    $b.validity$ = true
    add b to UpdateArray
    for each $e \in b$ **do**
      $e.validity$ = true
    end for
  end for
end for
clear $P+$
clear $P-$
clear C
for each b in UpdateArray **do**
  get K-nearest $p_1,\ldots p_k$ from P and M
  Compute $P(b) = \frac{\sum^{window} N_{valid}(P+M)}{\sum^{window} N(P+M)}$
  if $P(b) > b.Threshold$ **then**
    $b.validity$ = true
  else
    $b.validity$ = false
  end if
end for

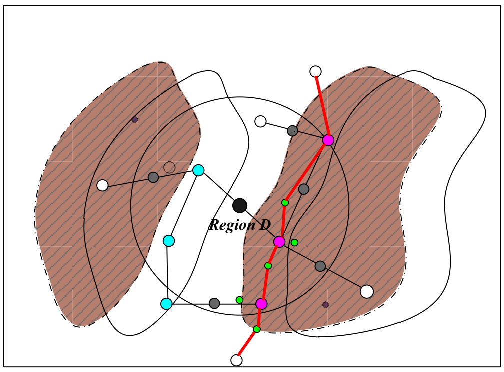

Fig. 4. Capacitance Boosting Strategy

do not change their validity. Three purple points belonging to $P+$ are boosted, and they activate different number of green incremental points. On the other side, three blue points are $P-$ points. As we have calculated in section III, the capacitance C_1 and C_2 are $3\varepsilon/d$ and $2\varepsilon/d$. Their normalization result are 0.65 and 0.45, respectively. Therefore, there are 3 and 2 (at most 5) boosting points activated around them. In addition, the other purple point in $P+$, which is not a simulated capacitor any more, has all incremental points closed to save calculation resource. The threshold of the predictive model is set to be $0.6-0.8$. Eventually, in Fig. 4, the red path with high safety will be searched in query phase. Actually, the positive half bridge area contains more paths because this region has been boosted. In other words, the positive electrode of simulated capacitors attracts paths to safe regions, while the negative electrode excludes paths to avoid unsafe regions. Details of the method described above are displayed in Algorithm 3.

V. EXPERIMENT AND DISCUSSIONS

For evaluating the proposed method, hundreds of simulation experiments are implemented in 3D workspace with two manipulators modeled by parameters of a practical 6-DOF Kawasaki FS03N manipulator. Two manipulators mounted on a fixed base make up a dual-manipulator system. Although it is a simple idea to plan two manipulators respectively, mutual collision avoidance and coordination between two manipulators are difficult to handle. Therefore, 12 DOFs of two manipulators are considered simultaneously and 12 dimensional C-space is constructed. The reachable workspace of two manipulators is decomposed into 406134 grids, and each grid is a cube with the size of 4x4x4 mm. Collision check in our system is implemented by a free 3D Collision Detection Library, ColDet 1.1. All experiments are carried out on an Intel Dual-Core 3.00 GHz CPU with 2GB memory. The experiment scenario is shown in Fig.5.

TABLE I
SAMPLING AMOUNT OF DIFFERENT METHODS

Method	P	M	M'	B	S	Time (s)
SCM	500	1537	-	2500	4537	4.02
CBB	500	1529	-	2500	4529	3.98
DBB	500	1549	466	2330	4379	3.21
DRM	4537	-	-	-	4537	5.38

Fig.5. shows the scenario with multiple obstacles, which indicates that the simulated capacitor method can be used in the multi-obstacle scene with different regions, so long as obstacles move with inertia. There are five bars in the experiment. One of them is vertical while the rest are horizontal. Since narrow passages in workspace often indicate presence and location of narrow passages in C-space [14], distances between obstacles and manipulators are set close enough to ensure difficulty. The vertical bar moves along the red axis back and forth, the highest bar moves left and right, while the other bars move up and down. All bars move at different speeds. Note that bars can pass through each other, while manipulators can not. Two manipulators are set to reach a

TABLE II

RESULTS OF DIFFERENT METHODS

Method	SPR	N_n	N_b	ART	LRT	SPT	AT	ST
SCM	91.51%	2092	55	34.21	52	403.2	0.123	49.59
CBB	91.74%	2900	871	33.12	54	401.0	0.241	96.64
DBB	91.66%	4379	2330	49.16	72	589.4	0.520	306.49
DRM	90.71%	4537	-	59.77	89	643.5	0.548	352.64
Improvement 1	0.80%	53.89%	-	42.76%	41.57%	37.34%	77.55%	85.93%
Improvement 2	-0.23%	27.86%	-	-3.20%	3.70%	-0.55%	48.96%	48.66%

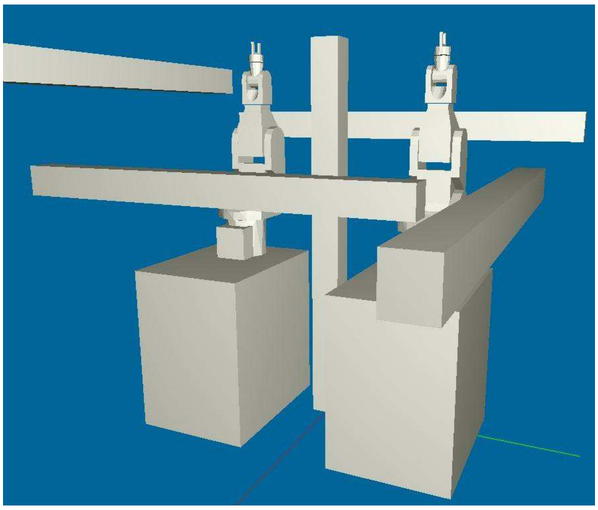

Fig. 5. Experiment Scenario

fixed destination from a random configuration for 500 times. Fig.6. shows a running state of the experiments. Results of different methods are shown in table II.

Table I compared the preprocessing phase of related methods: Simulated Capacitor Method (SCM), Capacitor Bridge Builder (CBB), Dynamic Bridge Builder (DBB) and DRM. The number of samples and sample time consuming are illustrated. The cardinality of P is crucial in realization. If it is too large, updating phase will be time-consuming. If it is too small, roadmap does not contain enough information for C-space construction. M is the number of middle nodes. S is the sum of sampling points, each main point generates K incremental points and edges, so parameter K influences the size of B and E. In this paper, K is set to be 5, contributing to a moderate point density. The method of DRM with equal total number of points to SCM is used for comparison. Column Time in Table I illustrates the preprocessing cost without W-C mapping.

In Table II, column SPT represents the Sum of Planning Times of each experiment, ART and LRT represents Average Re-searching Times and Largest Re-searching Times, respectively. SPR is the Successful Planning Rate, which is computed as $1-ART/SPT$. N_n is the average number of online nodes and N_b is the average number of online boosting nodes. AT is the Average planning Time of each planning, and ST is the average of Sum Time in each experiment.

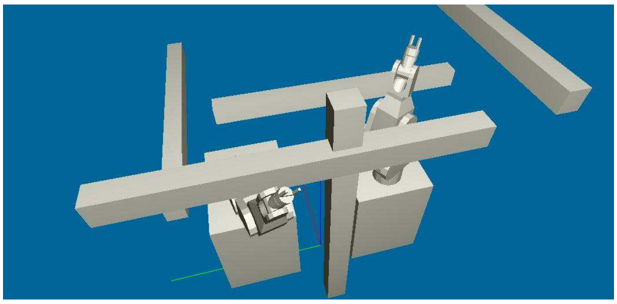

Fig. 6. A Running State in Experiment

As shown in Table II, the proposed work SCM and CBB show their superiority in SPT. Although these four different methods have almost the same SPR, considering the path safety, capacitor planner provides safer paths so that manipulators complete task with more than one third lower planning times. DBB [8] does well in difficult regions identification, but provides no safe guarantee. LRT and ART show its characters. Moreover, column N_n and N_b show the key point of the performance of SCM. Dealing with the same problem, SCM maintain a much smaller number of online nodes (less than half of DRM and DBB or two thrid of CBB) due to the capacitance boosting strategy. Clearly, closing the unnecessary incremental points can save lots of resource and concentrate on current difficult regions to achieve a shorter single and sum planing time. This point can be best illustrated by the smallest N_n, N_b, AT and ST in Table II. The first improvement row shows the efficiency improvement of SCM compared with DRM, and the last row shows the improvement compared with CBB method.

VI. CONCLUSIONS

In this paper, a simulated capacitor method is proposed to process the problem of difficult regions in changing environments. Difficult regions such as narrow passages are identified by the formed simulated capacitor between positive and negative toggled nodes in C-space. The capacitance is calculated with neighborhood information and reflects the local difficulty level. In the updating phase, incremental points, which are pre-sampled in preprocessing phase, are activated around positive poles of the capacitor where are safer and less likely disturbed by obstacles. As a result, the replanning times and total planning times are declined efficiently. What's more, considering the changing local difficulty level, incremental points will be regulated according to the capacitance, useless points will be shut down and attention will be concentrated on contemporary difficult regions, so that the sum number of points maintains acceptably and planning efficiency improves a lot. In our experiment, the superiority of this method in solving the difficult region problem is shown by high planning efficiency and low replanning times, which prove that the proposed simulated capacitor method is a promising method for path planning in changing environments.

REFERENCES

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

[2] R. Geraerts and M. H. Overmars, A comparative study of probabilistic roadmap Planners , *Proceedings of the Fifth International Workshop on the Algorithmic Foundations of Robotics*, pp. 249-264, 2002.

[3] D. Hsu, L. E. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin, On finding narrow passages with probabilistic roadmap planners. *In Proc. Int. Workshop on Algorithmic Foundations of Robotics*, pages 141-153, 1998.

[4] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, OBPRM: an obstaclebased PRM for 3D workspaces, *Proc. of the third International Workshop on the Algorithmic Foundations of Robotics*, pp. 155-168, 1998.

[5] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, MAPRM: a probabilistic roadmap planner with sampling on the medial axis o the free space, *IEEE International Conference on Robotics and Automation*, pp. 1024-1031, 1999.

[6] J. Denny, and N. M. Amato, Toggle PRM: Simultaneous Mapping of C-free and C-obstacle - A Study in 2D -, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2632-2639, 2011.

[7] M. Kalman, and M. Mataric, Motion planning using dynamic roadmaps, *IEEE International Conference on Robotics and Automation*, pp. 4399-4404, 2004.

[8] D. Ding, H. Liu, X. Deng, and H. Zha, A dynamic bridge builder to identify difficult regions for path planning in changing environments, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2925-2931, 2007.

[9] H. Liu, D. Ding, and W. Wan, Predictive Model for Path Planning by Using K-near Dynamic Bridge Builder and Inner Parzen Window, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2133-2138, 2008.

[10] K. I. Tsianos and L. E. Kavraki, Replanning: A powerful planning strategy for hard kinodynamic problems, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1667-1672, 2008. pp. 171-180, 1996.

[11] N. Chan, J. Kuffner, M. Zucker, Improved Motion Planning Speed and Safety using Regions of Inevitable Collision, In *ROMANSY*, pp. 103-114, 2008.

[12] K. Bekris and L. Kavraki, Greedy but safe replanning under kinodynamic constraints, *IEEE International Conference on Robotics and Automation*, pp. 704-710, 2007.

[13] S. Petti and T. Fraichard, Safe motion planning in dynamic environments, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2210-2215, 2005.

[14] J. P. van den Berg, and M. H. Overmars, Using work space information as a guide to non-uniform sampling in probabilistic roadmap planners, *IEEE International Conference on Robotics and Automation*, pp. 453-460, 2004.

[15] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, Choosing good distance metrics and local planners for probabilistic roadmap methods, *IEEE International Conference on Robotics and Automation*, pp. 630-637, 1998.

[16] J. Kuffner, Effective sampling and distance metrics for 3D rigid body path planning, *IEEE Inernational Conference on Robotics and Automation*, pp. 3993-3998, 2004.

[17] Hong Liu and Weiwei Wan, A subgoal-based path planner for unpredictable environment, *IEEE International Conference on Robotics and Automation*, pp. 994-1011, 2010.

[18] O. Gal, Z. Shiller and E. Rimon, Efficient and Safe On-Line Motion Planning in Dynamic Environments, *IEEE International Conference on Robotics and Automation*, pp. 88-93, 2009.

[19] J. Vannoy and J. Xiao, Real-time Adaptive Motion Planning (RAMP) of Mobile Manipulators in Dynamic Environments with Unforeseen Changes, *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1199-1212, 2008.

[20] Hong Liu and Tianwei Zhang, A “Capacitor” Bridge Builder Based Safe Path Planner for Difficult Regions Identification in Changing Environments, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3180-3186, 2012.