Path Planning for Robot-Assisted Active Flexible Needle using Improved Rapidly-Exploring Random Trees

Yan-Jiang Zhao, Felix Orlando Maria Joseph, Kaiguo Yan, Naresh V. Datla, Yong-De Zhang-*IEEE Member*, Tarun K. Podder-*IEEE & EMBS Member*, Parsaoran Hutapea, Adam Dicker, Yan Yu

***Abstract*—In robot-assisted needle-based medical procedures, path planning for a flexible needle is challenging with regard to time consumption and searching robustness for the solution due to the nonholonomic motion of the needle tip and the presence of anatomic obstacles and sensitive organs in the intended needle path. We propose a novel and fast path planning algorithm for a robot-assisted active flexible needle. The algorithm is based on Rapidly-Exploring Random Trees combined with reachability-guided strategy and greedy heuristic strategy. Linear segments are taken into consideration to the paths, and insertion orientations are relaxed by the introduction of the linear segments. The proposed algorithm yields superior results as compared to the commonly used algorithm in terms of computational speed, form of path and robustness of searching ability, which potentially can make it suitable for the real-time intraoperative planning for clinical procedures.**

I. Introduction

In minimally invasive surgeries, needle or probe insertion is probably one of the most pervasive procedures for diagnosis and therapies, such as tissue biopsies and radioactive seed implantations. However, the accurate and safe placement of the needle or probe for these procedures is challenging due to the anatomic obstacles and sensitive organs to be avoided. As an alternative to the traditional rigid needles and other passive flexible needles, we have been developing a flexible needle (non-beveled, symmetric tip) having an active or self-actuated capability. By utilizing the characteristic of shape memory alloys (SMA), the needle can generate a variety of curvatures of paths by utilizing different electric currents supplied to the SMA actuators [1].

In robot-assisted needle insertion procedures, path planning for the needle is a critical and challenging aspect for navigating the needle and robot to gain an accurate and safe operation. Moreover, steering a flexible needle in the soft tissue is a difficult problem due to the nonholonomic constraints of the needle tip, and the presence of anatomic obstacles and sensitive organs. In recent years, path planning for the passive flexible needles has been extensively studied in 2D and 3D environments by different approaches [2-11].

One of the common approaches for path planning is the mathematical formulation method. Alterovitz et al. formulated the path planning problem of a bevel tip flexible needle as a Markov Decision Process to maximize the probability of successfully reaching the target in 2D [2]. Duindam et al. formulated the problem as an optimization problem with an optimizing function and adopted discretization of the control space instead of the configuration space [3]. Park et al. proposed a path-of-probability algorithm to optimize the paths by computing a probability density function [4]. Our previous work presented a path planning algorithm based on multi-objective functions [5]. The mathematical methods formulate the problem as an optimization problem with objective functions, which are computationally expensive and may suffer from stability/convergence. Therefore, they are often used for preoperative planning.

Another important approach is the sampling-based method, such as Rapidly-Exploring Random Tree (RRT). Since Xu et al. [6] first applied an RRT-based method to search valid needle paths in 3D environments with obstacles, the RRT algorithm is now commonly used in the flexible needle path planning. Patil et al. utilized a modified version of the RRT method named Reachability-Guided RRTs [7], and then extended it to a dynamic environment using a replanning method [8]. Caborni et al. proposed a risk-based path planning for a steerable flexible probe using RRTs combining with the reachability-guided strategy and the goal bias strategy (RGGB-RRTs) [9]. Recently, Bernardes et al. proposed a fast intraoperative replanning algorithm in 2D and 3D environments based on RRT, which is similar to RGGB-RRTs

Research supported in part by the Department of Defense CDMRP Prostate Cancer Research Program (Grants# W81XWH-11-1-0397/98/99), by National Natural Science Foundation of China (Grant # 51305107) and by Science and Technology Project of Heilongjiang Provincial Education Department, China (Grant# 12531110).

Yan-Jiang Zhao is with the Department of Radiation Oncology, Thomas Jefferson University, Philadelphia, PA 19107 USA, and is with the Intelligent Machine Institute, Harbin University of Science and Technology, Harbin, Heilongjiang 150080 China (e-mail: Yanjiang.Zhao@jefferson.edu; zhaoyj@hrbust.edu.cn).

Felix Orlando Maria Joseph is with the Department of Radiation Oncology, Case Western Reserve University, Cleveland, OH 44106 USA (email: fom@case.edu).

Kaiguo Yan is with the Department of Radiation Medicine, Medstar Georgetown University Hospital, Washington, DC 20007 USA (e-mail: Kaiguo.Yan@gunet.georgetown.edu).

Naresh V. Datla is with the Department of Mechanical Engineering, Temple University, Philadelphia, PA 19122 USA (e-mail: datla@temple.edu).

Yong-De Zhang is with the Intelligent Machine Institute, Harbin University of Science and Technology, Harbin, Heilongjiang 150080 China (e-mail: zhangyd@hrbust.edu.cn).

Tarun K. Podder is with the Departments of Radiation Oncology, University Hospitals, Case Western Reserve University, Cleveland, OH 44106 USA (phone: 216-844-2580; fax: 216-844-2005; e-mail: tarun.podder@ case.edu).

Parsaoran Hutapea is with the Department of Mechanical Engineering, Temple University, Philadelphia, PA 19122 USA (e-mail: hutapea@temple.edu).

Adam Dicker is with the Department of Radiation Oncology, Thomas Jefferson University, Philadelphia, PA 19107 USA (e-mail: adam.dicker@jeffersonhospital.org).

Yan Yu is with the Department of Radiation Oncology, Thomas Jefferson University, Philadelphia, PA 19107 USA (e-mail: Yan.Yu@jefferson.edu).

[10-11]. The main advantage of the RRT method is that it is fast and easy to implement. However, the probabilistic nature of the RRT algorithm compels it to find a feasible path which may or may not be the globally optimal solution [12].

To summarize the existing path planning algorithms, firstly, in the flexible needle path planning, all the algorithms are only aiming at utilizing the curvilinear paths, but not considering the linear segments, which may both shorten the length of path and save the cost of control and energy for the active needle (you do not have to make the needle bent by actuators). Although Patil et al. relaxed the curvatures of the curvilinear paths which allowed the linear segments in the paths theoretically, because of the probabilistic nature of the RRT algorithm, the possibility for the appearance of the linear segment is nearly non-existent [7-8]. Secondly, most of the algorithms, if not all, are with the routine method that the insertion orientation is fixed or specified, e.g. to be orthogonal to the skin surface, therefore the planning or optimizing results are constrained originally. Although Xu et al. relaxed the insertion orientation by a back-chaining method, however, the orientation of approaching to the goal is fixed originally [6].

In this paper, a novel and fast path planning algorithm based on RRT, combining the reachability-guided strategy and the greedy heuristic strategy, for a self-actuated flexible needle is proposed. It has been named Reachability and Greedy Heuristic Guided RRTs (RGHG-RRTs). We have adopted the variable but bounded curvatures, and more importantly we have taken account of the linear segments and the relaxation of insertion orientations to the needle paths.

II. Path Planning Algorithm

A. Outline of RGHG-RRTs Algorithm

The workflow is as shown in Algorithm 1. Unlike the previous RRT algorithms, once initialized with q_{init}, this algorithm does not immediately generate a random node, instead, it checks whether a path can be generated directly from the initial node q_{init} to the goal node q_{goal} including linear or curvilinear collision-free paths (in line 2-8). LinearCheck() is to check whether a linear segment can be generated between two points without colliding with the obstacles. GeneratePath() is to extract the paths connecting q_{init} and q_{goal} by linear or curvilinear segment(s). If the linear path is generated, the searching is over because it is obviously the best path. If not, it checks for a curvilinear path. If both results of Reachable() and Valid() functions are true, it will generate a curvilinear path directly from q_{init} to q_{goal}. Reachable() is to check if q_{goal} is in the reachable region of the needle insertion from q_{init} under the nonholonomic constraints (detailed statements are in the following Reachability-guided strategy part). Valid() is to check if the whole path is in the collision-free space $\boldsymbol{Q}_{free}$. And then the algorithm goes into the loop routine section. The algorithm begins to generate a new node q_{new} randomly by the routine RandomNode() in line 10, which is imbedded with a function CollisionCheck(), if a new randomly generated node q_{rand} is in collision with obstacles, it will be abandoned until a new valid random node is generated and named as q_{new}. And then it searches for trees and paths in a greedy heuristic way and iterates until the terminate condition is reached.

Algorithm 1: RGHG-RRTs (q_{init}, q_{goal}, *max_path*)

```
 1: Trees: Initialization ($q_{init}$)
 2: if LinearCheck($q_{init}$, $q_{goal}$, $\boldsymbol{Q}_{obs}$)
 3:    path=GeneratePath ($q_{init}$, $q_{goal}$, $\boldsymbol{u}$)
 4:    return
 5: end
 6: if Reachable ($q_{init}$, $q_{goal}$, $r_{min}$)&&Valid ($q_{init}$, $q_{goal}$, $\boldsymbol{Q}_{obs}$, $\boldsymbol{u}$)
 7:    path=GeneratePath ($q_{init}$, $q_{goal}$, $\boldsymbol{u}$)
 8: end
 9: while (path_number<max_path)&&(iteration<max_iteration)
10:    $q_{new}$=RandomNode (CollisionCheck($q_{rand}$, $\boldsymbol{Q}_{obs}$))
11:    if LinearCheck($q_{init}$, $q_{new}$, $\boldsymbol{Q}_{obs}$)
12:       tree=GenerateTree ($q_{init}$, $q_{new}$, $\boldsymbol{u}$)
13:       Trees.add_tree
14:       if Reachable ($q_{new}$, $q_{goal}$, $r_{min}$)&&Valid ($q_{new}$, $q_{goal}$, $\boldsymbol{Q}_{obs}$, $\boldsymbol{u}$)
15:          path=GeneratePath (Trees, $q_{goal}$, $\boldsymbol{u}$)
16:          Paths.add_path
17:       end
18:    end
19:    if Reachable ($q_{init}$, $q_{new}$, $r_{min}$)&&Valid ($q_{init}$, $q_{new}$, $\boldsymbol{Q}_{obs}$, $\boldsymbol{u}$)
20:       tree=GenerateTree ($q_{init}$, $q_{new}$, $\boldsymbol{u}$)
21:       Trees.add_tree
22:       if Reachable ($q_{new}$, $q_{goal}$, $r_{min}$)&&Valid ($q_{new}$, $q_{goal}$, $\boldsymbol{Q}_{obs}$, $\boldsymbol{u}$)
23:          path=GeneratePath (Trees, $q_{goal}$, $\boldsymbol{u}$)
24:          Paths.add_path
25:       end
26:    end
27:    $q_{proper}$=FindProperNode (Trees, Reachable($q_{old}$, $q_{new}$, $r_{min}$))
28:    if Valid ($q_{proper}$, $q_{new}$, $Q_{obs}$, $\boldsymbol{u}$)
29:       tree=GenerateTree ($q_{proper}$, $q_{new}$, $\boldsymbol{u}$)
30:       Trees.add_tree
31:       if Reachable ($q_{new}$, $q_{goal}$, $r_{min}$)&&Valid ($q_{new}$, $q_{goal}$, $\boldsymbol{Q}_{obs}$, $\boldsymbol{u}$)
32:          path=GeneratePath (Trees, $q_{goal}$, $\boldsymbol{u}$)
33:          Paths.add_path
34:       end
35:    end
36: end
37: return Paths
```

B. Greedy Heuristic Strategy

In contrast to all the previous RRT algorithms, we propose a greedy heuristic strategy. On the one hand, it is greedy for initial node to generate new trees; on the other hand, it is greedy for the goal node to achieve paths. After q_{new} is obtained, instead of adding it to an existing tree in the previous algorithms, our proposed algorithm will greedily check whether a linear segment could be generated connecting q_{init} to q_{new}, if yes, a new tree starting with a linear segment will be generated (line 11-13). By doing so, it successfully relaxes the insertion orientations, which can be in various angles rather than just orthogonal to the skin surface. It is observed that q_{new} is directly the candidate node to extend the trees [9]. If the tree is generated, it goes on to verify whether q_{new} could connect q_{goal} (line 14-17), if yes, a new path will be gained. Here we are greedy for searching the goal, because we believe that the direct connection from q_{new} to q_{goal} is probabilistically superior to those which travel round and then connect to the goal. Then it will search for a curvilinear path with the similar process (line 19-26).

If no path is gained after above search operations, it will try to extend the other trees that have not achieved a path (line 27-35). In contrast to searching for a nearest node in the previous algorithms, it searches for a proper node (by routine 27). The idea of proper node means the node should not be too near to an existing node q_{old}, the distance should be larger than a specific metric ρ in order to prevent the insufficient growth.

C. *Reachability-Guided Strategy*

Because of the presence of the nonholonomic constraints, it is possible that a node is not reachable for the specified configuration q_{spec}. The reachable region of the needle is a leaflike area as shown in Fig. 1. There are two regions of ***A*** and ***B*** that cannot be reached no matter how deeply the needle is inserted or bent considering a realistic design of a needle. In order to speed up the search and make the following computation efficient and effective, we have checked the reachability of the objective node (may be q_{new} or q_{goal}). In the local frame $\{L\}$ at q_{spec}, the reachable region is defined by [9]

$$y_L \geq \sqrt{2r_{\min}\,|\,x_L\,| - x_L^2} \tag{1}$$

where x_L, y_L are the coordinates of the objective node in the frame $\{L\}$, $r_{\min}$ is the minimum radius constraint for the active flexible needle.

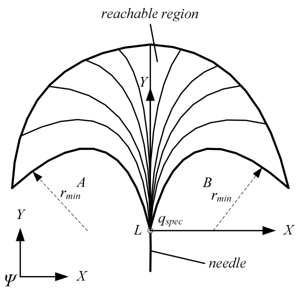


Fig. 1 Reachable region of the active flexible needle

D. *Control Input Solving*

If the objective node lies in the reachable region, the algorithm will calculate the inputs and generate a branch to the tree. We formulate the node as q: (x; y; θ), where x, y are the coordinates of the node, and θ is the deviation angle of the needle tip direction from axis y, all the parameters are with respect to the world frame $\{\Psi\}$. The motion of the needle tip is controlled by two inputs: the insertion speed v(t) and the electric current I(t), so the inputs ***u*** can be formulated as ***u***: (v; I). Because the inputs drive the needle to perform a geometric trajectory, we can encode the path with geometric parameters instead of the actual control inputs ***u***, avoiding the inefficient performance that randomly samples control inputs to compute the best combination. The path is composed of series of segments $\{S_1, S_2, \ldots, S_n\}$ [7], each of which can be parameterized as u_i: (l_i; r_i), and can be computed in the local frame $\{L\}$ as follows: (2) is for a linear segment, (3) is for a curvilinear segment.

$$\begin{cases} l_i = \sqrt{x_i^2 + y_i^2} \\ r_i = \infty \\ \varphi_i = -\arctan(x_i / y_i) \end{cases} \tag{2}$$

$$\begin{cases} r_i = (x_i^2 + y_i^2) / |\,2x_i\,| \\ \varphi_i = \pi - 2\,|\arctan(y_i / x_i)\,| \\ l_i = r_i \cdot \varphi_i \end{cases} \tag{3}$$

where l_i, r_i are the length and the radius of the ith segment, respectively; x_i, y_i are the local coordinates of the objective node in the local frame $\{L\}$; φ_i is the deviation angle of the needle tip direction from axis y in the local frame $\{L\}$. Then the configuration θ_i in the world frame $\{\Psi\}$ is defined as:

$$\theta_i = \begin{cases} \theta_{i-1} - \varphi_i, & x_i > 0 \\ \theta_{i-1} + \varphi_i, & x_i < 0 \end{cases} \tag{4}$$

E. *Optimizing Function*

Among the candidate paths under optimization, all of which have already met the required constraints, the optimal path can be chosen based on a cost function as:

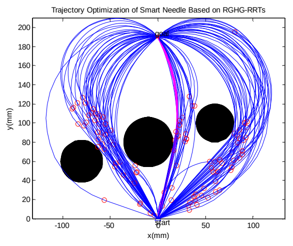
$$\min F(\boldsymbol{u}, T) = \min\{\alpha_1 L + \alpha_2 D + \alpha_3 S + \alpha_4 N\} \tag{5}$$

where L is the length of the path; D is the degree of danger in the path, relative to the distance between the path and the obstacles; S is the curve valuation of the path, evaluated by curvatures, which is relative to the force-torque on the needle as well as temperature rise; N is the degree of control, i.e. the number of segments of the whole path which is relative to the control cost; α_1~α_4 are the weighted coefficients. The result of the function is the overall evaluation of the optimal path.

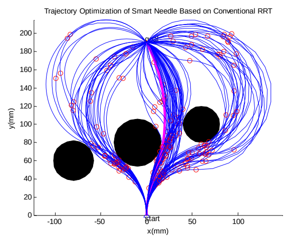
III. RESULTS

A. *Test Case 1*

In this paper, we focused on a 2D environment. We firslty set the scale of the envirnment to 200mm×200mm, and the minmum radius $r_{\min}$=50mm [7]. The specific metric ρ=10mm. The weighted coefficients α_1=α_3=α_4=1. Assuming that the obstacle is containing a relative belt of safe margin around it, in order to speed up the computation, we can disregard the second term in (5) by setting α_2=0. The maximum number of the candidate paths is set to 100, and the maximum number of iterations is set to 1000. In order to compare the performance of our proposed RGHG-RRTs algorithm with the commonly used RGGB-RRTs algorithm, we also set the RGGB-RRTs algorithm into the same environment except that the maximum number of iterations is set to 5000, otherwise it would fail to get enough candidate paths.We have simulated the motion planner in MATLAB® (ver. 7.8.0, R2009a; MathWorks, Natick, MA) on a 2.5 GHz 4-core Intel® i5™ PC. Fig. 2 shows the results of the two approaches for one optimization, respectively. We have also tried 50 times and achieved all the optimal results of the above mentioned two approaches as presented in Fig. 3 as well as in Table I. In the figures, the black circles are the obstacles, the small red circles are the conjunction points of the two segments in the paths. In the tables, the results from the second to the fourth columns are the "mean ± standard deviation" of the 50 simulations.

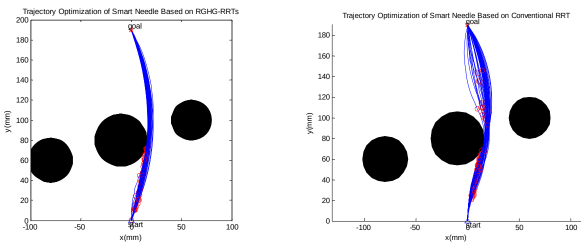


(a) RGHG-RRTs algorithm



(b) RGGB-RRTs algorithm

Fig.2 Optimization results. The pink line is the optimal path.



(a) RGHG-RRTs algorithm (b) RGGB-RRTs algorithm
Fig. 3 Optimal results of the 50 simulations

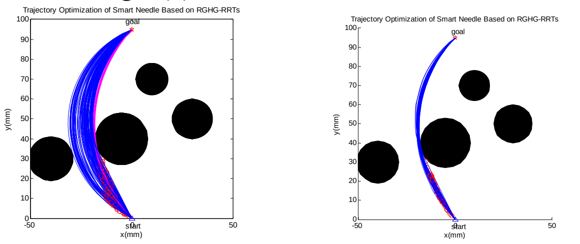
TABLE I. COMPARISON BETWEEN THE TWO APPROACHES

Approaches	CPU time for one tree (sec)	Value of function	No. of iterations
RGHG-RRTs	0.0013±0.0002	196.98±0.80	429±38
RGGB-RRTs	0.0467±0.0230	198.68±0.96	3805±399

From the results, we can conclude that: (1) the RGHG-RRTs algorithm can effeciently achieve variety of the paths with linear segments as well as curved segments, and the optimal path was likely to be the one with linear segments; (2) the RGHG-RRTs algorithm was superior to the RGGB-RRTs algorithm thoroughly, the speed was 35 times faster, the optimial path was always better (the less the value of function, the better the path), and the number of iterations was much smaller.

B. *Test Case 2*

In order to test and compare the two algorithms further, we have created a more complex environment by compressing the environment and adding an extra obstacle. We have shrunk the environment to half and then added an additional obstacle. The maxmum iteration number is set to 10000. Other environments are just the same as in test case 1. While the RGGB-RRTs algorithm failed to obtain any solution, The RGHG-RRTs algorithm successfully and rapidly achieved 100 trajectories and the optimal one was calculated. The result is shown in Fig. 4(a). We have also simulated 50 times of the opimizations, the results are presented in Fig. 4(b) and Table II.



(a) Result of one optimization (b) Results of 50 optimizations
Fig. 4 Result of RGHG-RRTs

TABLE II. RUSULTS OF RGHG-RRTS

Approaches	CPU time for one tree (sec)	Value of function	No. of iterations
RGHG-RRTs	0.0181±0.0019	106.72±0.56	7051±712

Results showed that the RGHG-RRTs algorithm provided a more possibility for an optimal solution by relaxing the insertion orientations. Although the searching efficiency suffered from the complexity of environment compared to Test Case 1, it was still very fast.

IV. CONCLUSION

We have proposed a fast path planning algorithm named RGHG-RRTs algorithm which is developed based on RRT for a robot-assisted active flexible needle steering. We have formulated a greedy heuristic strategy and combined it with the reachability-guided strategy to speed up the search and to improve the convergence. We have adopted the linear segments to the paths, and the insertion directions are relaxed by the introduction of the linear segments. We have also formulated an optimizing function, by which the optimal path can be achieved from the sub-optimal candidate paths. Simulation tests were done in 2D environments with obstacles. In comparison with the RGGB-RRTs algorithm, the performance of the proposed RGHG-RRTs algorithm was superior in terms of computational speed, form of path and robustness of searching ability. In the future work, we will extend this algorithm into a 3D dynamic envrionment to achieve the real-time intraoperative planning for clinical operations.

REFERENCES

[1] T. K. Podder, A. P. Dicker, P. Hutapea, K. Darvish, Y. Yu, "A novel curvilinear approach for prostate seed implantation," *Med Phys*, vol. 39, no. 4, pp. 1887-1892, Mar. 2012.

[2] R. Alterovitz, M. Branicky, K Goldberg, "Motion planning under uncertainty for image-guided medical needle steering," *International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1361-1374, Nov. 2008.

[3] V. Duindam, R. Alterovitz, S. Sastry, K. Goldberg, "Screw-based motion planning for bevel-tip flexible needles in 3D environments with obstacles," in *IEEE Int. Conf. on Robotics and Automation*, Pasadena, 2008, pp. 2483-2488.

[4] W. Park, Y. Wang, G. S. Chirikjian, "The path-of-probability algorithm for steering and feedback control of flexible needles," *International Journal of Robotics Research*, vol. 29, no. 7, pp. 813-830, Jun. 2010.

[5] Y. J. Zhao, Y. D. Zhang, F. Tu, "Reverse path planning for flexible needle in 2D soft tissue with obstacles," *Applied Mechanics and Materials*, vol. 121-126, pp. 4132-4137, Oct. 2011.

[6] J. Xu, V. Duindam, K. Goldberg, "Motion planning for steerable needles in 3D environments with obstacles using rapidly exploring random trees and backchaining," in *IEEE Int. Conf. on Automation Science and Engineering*, Washington, 2008, pp. 41-46.

[7] S. Patil, R. Alterovitz, "Interactive motion planning for steerable needles in 3D environments with obstacles," in *IEEE RAS/EMBS Int. Conf. Biomedical Robotics and Biomechatronics*, Tokyo, 2010, pp. 893-899.

[8] S. Patil, J. Burgner, R. J. Webster III, R. Alterovitz, "Needle steering in 3-D via rapid replanning," *IEEE Trans. on Robotics*, to be published.

[9] C. Caborni, S. Y. Ko, E. D. Momi, et al., "Risk-based path planning for a steerable flexible probe for neurosurgical intervention," in *IEEE RAS/EMBS Int. Conf. Biomedical Robotics and Biomechatronics*, Rome, 2012, pp. 866-871.

[10] M. C. Bernardes, B. V. Adorno, P. Poignet, G. A. Borges, "Robot-assisted automatic insertion of steerable needles with closed-loop imaging feedback and intraoperative trajectory replanning," *Mechatronics*, vol. 23, no. 6, pp. 630-645, Sep. 2013.

[11] M. C. Bernardes, B. V. Adorno, G. A. Borges, P. Poignet, "3D robust online motion planning for steerable needles in dynamic workspaces using duty-cycled rotation," *Journal of Control, Automation and Electrical Systems*, to be published.

[12] S. Karaman, E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846-894, Jun. 2011.