2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Motion planning for gantry mounted manipulators: A ship-welding application example

Anders Lau Olsen & Henrik Gordon Petersen
The Maersk Mc-Kinney Moller Institute
University of Southern Denmark
Niels Bohrs Alle 1, 5230 Odense M, Denmark
Email: {alauo,hgp}@mip.sdu.dk

Abstract— **We present a roadmap based planner for finding robot motions for gantry mounted manipulators for a line welding application at Odense Steel Shipyard (OSS). The robot motions are planned subject to constraints on when the gantry may be moved. We show that random sampling of gantry configurations is a viable technique for positioning the manipulator and present a pruning technique for managing the growth of the roadmap. We discuss results from simulations and from applications at the shipyard, where a similar planner has now been implemented for production.**

I. Introduction

Steel shipyards commonly apply robots for the welding of blocks of ships. The blocks are put together on an assembly line and moved into the working area of the robots. The robots are typically constructed from 5-6 degrees of freedom (*dof*) manipulators mounted on 1-3 *dof* gantries. For the welding of short lines, the gantry serves as a positioning device for the manipulator while for long lines both the gantry and the manipulator may move.

Robot motions are calculated off-line, but are adjusted on-line to adapt to inaccuracies of the block and the robot. The tip of the welding tool works as a sensor by the help of which the precise position of a weld line can be measured: The tool is placed near the weld line and moved in a number of pre-planned directions; each move is stopped instantaneously when the tool tip makes contact with a steel plate. This measurement process is called a *sensing*.

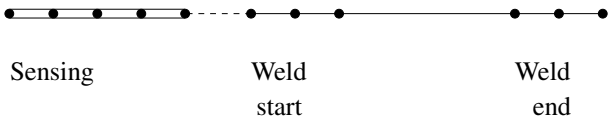
Movement of the gantry is typically less accurate than movement of the manipulator alone. Therefore, to achieve a precise calibration, the gantry should not be moved during a sensing. Also for some installations, starting and stopping of the gantry may cause the base of the manipulator to oscillate. For reasons of welding quality, one should therefore try to ensure that the gantry has no sudden accelerations. If possible the gantry should be kept fixed throughout the welding of a line.

Our planner for this application combines probabilistic roadmap (PRM) based motion planning [1] with a randomized algorithm for gantry positioning. The motion planner uses shortest–path searching of the roadmap combined with deferred verification of roadmap paths [2], [3], [4]. The constraint that the tool must follow the paths of the sensings and weldings is incorporated in the roadmap algorithm by the technique of Han & Amato [5]. The gantry positioning algorithm calculates an approximation of the region within which to position the gantry if a set of targets are to be reached and then randomly samples this region. Seraji [6] makes related geometrical considerations, and Cortés [7] uses also a workspace approximation to guide a random sampling. Our sampling based algorithm stands in contrast to the more common iterative optimization approaches to the base positioning problem [8], [9], [10].

II. Planning algorithm

The planner is written for 5-6 *dof* manipulators mounted on 3 *dof* gantries. We use the term *robot* to refer to the full system of gantry and manipulator.

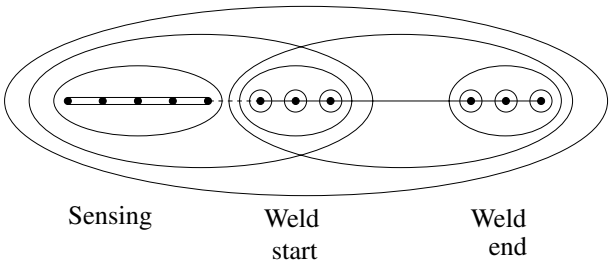
To illustrate the path planning algorithm, consider a simple job consisting of a sensing followed by a welding. The sensing is described by a sequence of targets along which to move the tool. The description of the welding movement can be split into a set of targets for the tool movement at the start of the weld line and a set of targets for the end of the weld line. The structure of the simple job is shown in Fig. 1.



1: A job with a single sensing and welding.

Targets of a job are put into *groups* according to application requirements. A target can be in multiple groups, and a group need not cover a consecutive sequence of targets. Each group represents a part of the job where it is *beneficial* that the gantry position stays the same. Innermost groups represent parts where the gantry is moreover *required* to stay the same. More precisely, the gantry may change position in between a pair of targets only if the line connecting these targets crosses the border of a group.

Fig. 2 shows a sensible grouping for the welding job of Fig. 1. The gantry may not be moved during the sensing, and therefore the sensing targets are put in the same innermost group. Each weld target is in an innermost group of its own



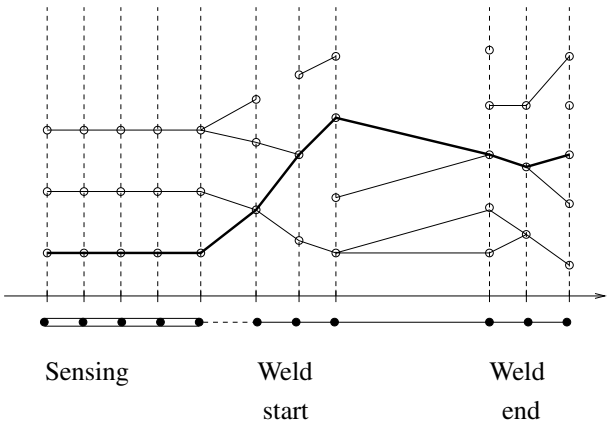
2: Grouping of targets of a job.

(the smallest circles in the figure), and therefore movement of the gantry in between weld targets is permitted. The larger groups signify that it is beneficial for the gantry to be kept fixed during for example the start of the weld, the end of the weld, the entire weld line, or the entire job.

The planner assumes that closed–form inverse kinematics (IK) for the 5-6 *dof* manipulator is known. The closed–form formula yields a finite number of solutions where each solution corresponds to a style of posture of the manipulator. The IK procedure called by the planner returns for each IK solution also the style of posture.

The robot is viewed as a closed–chain kinematic system: The open chain of gantry and manipulator is closed by the requirement that the tool follows the sensing and welding paths. The *configuration* of the robot is a vector containing the joint values for the robot and values for the position of the tool on the sensing or welding path. The configuration is split into an *active* part for which motion planning is done, and a *passive* part for which joint values are found found by the IK procedure [5]. The active part covers the gantry and the position of the tool on its path, and the passive part covers the manipulator.

Motion planning is done via a roadmap with nodes and edges. Each node corresponds to a configuration, and each edge represents a motion that connects the pair of configurations. A configuration for a node is computed by sampling a value for the active part. The passive part of the configuration is computed by the IK procedure. Motions for edges are computed by linear interpolation on the active part. Nodes are inserted in the roadmap only for the tool positions that correspond to targets of the job. This simplifies the roadmap construction and the type of the resulting motions, but implies also that the planning algorithm is *incomplete* in the sense that the planner searches through only a subset of of the possible robot movements. Also a node of a target is connected only to nodes of the next and previous target. According to the rules for groups, a node of a target can be connected to a node of a neighbor target only if the nodes have the same gantry position, or if the line between the targets crosses the border of a group. A roadmap in this style is shown in Fig. 3. The roadmap has a layer for each target of the job. A path leading from the first to the last layer is a candidate path for a motion for the job. The tool may not move backwards on its path, and therefore the candidate path must lead through the layers from left to right in order.



3: A layered roadmap with a candidate path for a motion for the job.

The path planner incrementally expands the roadmap (Section IV) and then searches the roadmap for a valid path (Section V). Following these two steps, the roadmap is subject to a refinement step (Section VI) to control the growth of the roadmap. The steps are repeated as long as time permits to allow the search step to find better and better paths. The planner thus doesn't just find a solution, but iteratively improves the solution. A key part of the planning algorithm is the randomized procedure for positioning the gantry in such a way that the manipulator is likely to be able to reach all targets of a group. This procedure is described in Section III.

III. Gantry positioning

Given a group of targets G we wish to randomly select a gantry position from the subset $\mathcal{W}_G$ of the gantry workspace $\mathcal{W}_{gantry}$ for which all targets of the group can be reached. For $g \in G$ let $\mathcal{W}_g$ be the set of gantry positions that are reachable when the tool of the manipulator has been attached to g. We call this region the *target region* for g. Intuitively, to reach all targets the gantry should be positioned within the intersection of these regions:

$$\mathcal{W}_G = \bigcap_{g \in G} \mathcal{W}_g$$

To arrive at a simple to implement and practical algorithm, we approximate each target region by a ball with an effective radius R. To make the approximation conservative, the radius R should be an upper bound for the maximum reach of the manipulator. To simplify matters further, the only targets considered are those that are supporting points for the smallest enclosing ball for all the targets. (The subset of at most 4 supporting points is defined as a smallest subset of targets that has the same smallest enclosing ball as all of the targets.) Let $\mathcal{B}_1, \ldots, \mathcal{B}_k$ be the balls with radius R centered at these supporting points. We may then easily compute the smallest enclosing ball $\mathcal{B}_e$ for $\bigcap_{i=1}^{k} B_i$. Let now in general $P(\mathcal{B})$ be the smallest box with the same orientation as the box $\mathcal{W}_{gantry}$ that encloses the intersection $\mathcal{W}_{gantry} \cap \mathcal{B}$. The

gantry position can then be randomly sampled from:

$$\mathcal{W}_{G',1} \equiv P(\mathcal{B}_e) \tag{1}$$

or alternatively from the intersection of boxes:

$$\mathcal{W}_{G',2} \equiv \bigcap_{i=1}^{k} P(\mathcal{B}_i) \tag{2}$$

Fig. 4 illustrates the derivation of the approximations of $\mathcal{W}_G$. gantry. Fig. 4 (a) shows four targets $G = \{g_1, \ldots, g_4\}$ placed in the plane of a two-dimensional gantry. For each target g the target region $\mathcal{W}_g$ is illustrated by a closed curve. The targets may have different orientations, and therefore the regions $\mathcal{W}_g$ may differ in shape. To reach all targets, the gantry should be positioned in the intersection $\mathcal{W}_G$ of the four target regions. Fig. 4 (b) shows the approximation $\mathcal{W}_{G',2}$ of $\mathcal{W}_G$. The smallest ball enclosing G has supporting points at g_1 and g_4. The target regions for g_1 and g_4 are approximated by the balls $\mathcal{B}_1$ and $\mathcal{B}_2$ whose intersection is enclosed by the ball $\mathcal{B}_e$. $\mathcal{W}_{G',1}$ is the box enclosing the intersection of $\mathcal{B}_e$ and $\mathcal{W}_{gantry}$. Fig. 4 (c) shows the approximation $\mathcal{W}_{G',1}$ of $\mathcal{W}_G$. The intersection of $\mathcal{B}_1$ and $\mathcal{B}_2$ with $\mathcal{W}_{gantry}$ is enclosed in the boxes $P(\mathcal{B}_1)$ and $P(\mathcal{B}_2)$. The gantry position is sampled from the intersection $\mathcal{W}_{G',2}$ of these two boxes.

IV. Roadmap expansion

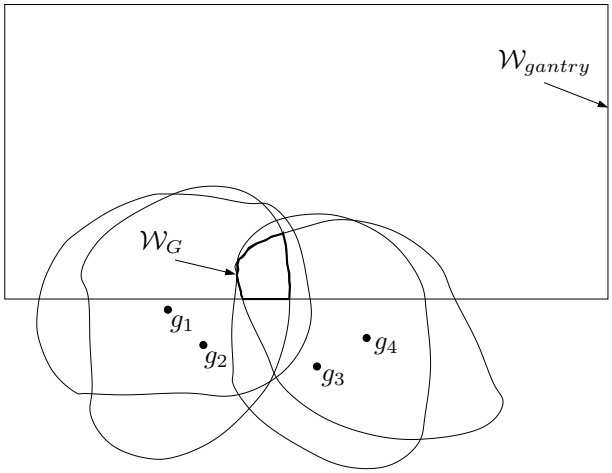
The growing of the roadmap is guided by the grouping of targets of the job. For each group G the planner samples a gantry position from the volume $\mathcal{W}_{G',1}$ or $\mathcal{W}_{G',2}$ described in Section III. The planner then calls the IK procedure to verify if the targets of G can all be reached from this gantry position. We wish to find targets where the IK procedure fails as quickly as possible. We therefore apply a heuristic, where the next target the IK procedure is called for is the target that is furthest away from the previously checked targets.

The planner checks that the targets are reached for the same style of manipulator posture and that joint speed limits are not violated. No collision detection for the IK solutions is done at this stage. If all the checks succeed, nodes for the IK solutions are inserted in the roadmap and connected to neighbor nodes. A node is connected to a neighbor node only if the rules for posture, joint speed limits, and change of gantry position are satisfied.

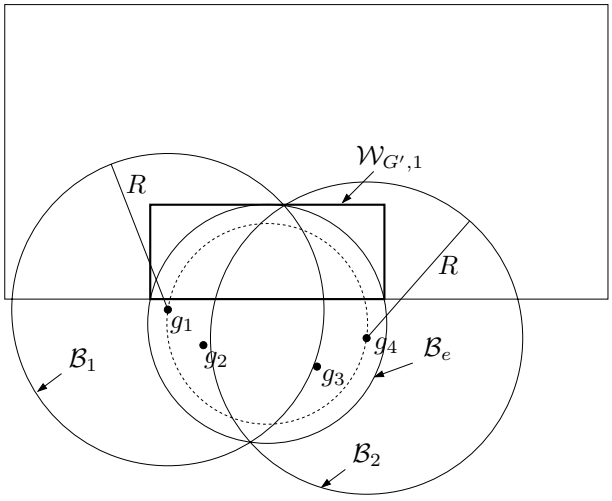
The planner makes only a single gantry positioning attempt for each group. The graph grows therefore by only a small amount for each roadmap expansion step. For the easy small groups, nodes will be inserted often, while for other groups nodes may rarely or never be inserted. The steps of the planner should therefore be iterated for long enough that nodes for the larger and more difficult groups are likely to have been inserted also.

V. Roadmap searching

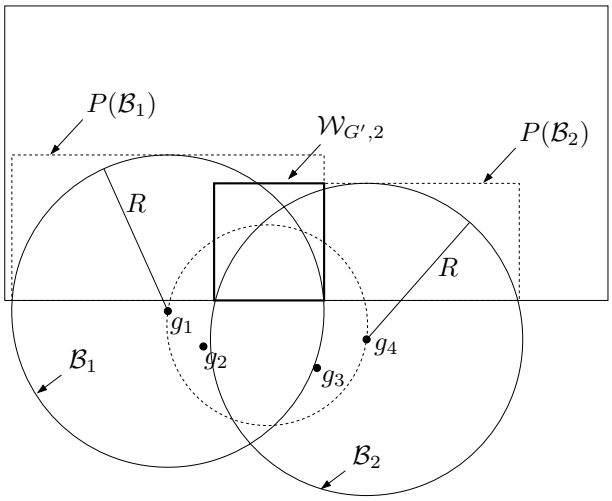
Following the roadmap expansion step, a search for the shortest path through the roadmap is done. The shortest–path searching is implemented with Dijkstra's algorithm. Due to the simple, layered structure of the roadmap (Fig. 3), no



(a) The exact region $\mathcal{W}_G$ for the positioning of the gantry with respect to the group of targets $G = \{g_1, \ldots, g_4\}$.



(b) Approximation of $\mathcal{W}_G$ by (1).



(c) Approximation of $\mathcal{W}_G$ by (2).

4: Two-dimensional derivation of regions for the sampling of gantry positions.

priority queue is needed in Dijkstra's algorithm, and the search runs in time proportional to the number of nodes and edges.

The cost of an edge is measured by a configuration space metric and by whether the gantry changes position or not. Avoiding the change of gantry position is given precedence over the configuration space metric. With this cost measure, the path planning algorithm will for a simple job (Fig. 1) typically find either a solution where the gantry is kept fixed throughout or a solution where the gantry is moved exactly once, namely on the move from the start to the end of the weld line.

Once a shortest path has been extracted from the roadmap, the planner must verify if robot can actually traverse the path. The planner first checks all nodes of the path for collisions. The planner keeps track of the failure rate of collision checking for targets. The nodes for the targets for which no collision checking has been done and the targets with the highest failure ratio are tested first. Using the IK procedure the planner then checks if the robot kinematically is able to traverse the edges of the path. Finally, collision checking for the edges is done. The planner applies the technique [4], [11] of always checking the center of the longest edge subsegment for which no checks have yet been done.

If verification for a node or edge fails, it is removed from the roadmap. The shortest–path search is repeated until either a valid path is found, or the roadmap becomes disconnected.

VI. Roadmap refinement

Naïve expansion of the roadmap (Section IV) followed by path extraction and verification (Section V) rapidly results in a large roadmap. Relative to the cost of maintaining their presence in the roadmap, the many nodes contribute little to the chance of finding a valid path through the roadmap: Some of the nodes are never part of a shortest path, because for example their posture is wrong. Other nodes are occasionally on a shortest path, but are never checked for collisions because other nodes on the path are always found to be in collision first.

To manage the growth of the roadmap we perform a pruning step: Check a small percentage of the nodes of the roadmap for collisions and remove those found to collide. Let N_G be the set of nodes inserted for the group G during the previous roadmap expansion step. If none of the nodes of N_G are known to be collision free, then remove all nodes of N_G. Otherwise leave the nodes of N_G in the roadmap. The pruning step allows us to have deferred collision checking together with a roadmap that only slowly grows over time. The nodes of N_G have the same gantry position and style of posture. The pruning step therefore tends to keep unchecked nodes in the roadmap that are likely to be collision free.

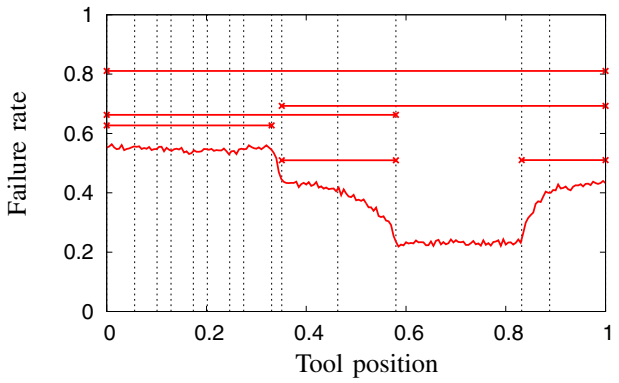
Aside from pruning the roadmap, it pays off to also selectively grow the roadmap within difficult parts of the job. The difficulty of each part of the job is measured by the number of collision free nodes of a target. The roadmap is grown at the most difficult target g by the roadmap expansion for a group (Section IV) except that nodes for the group are inserted only if the node for g is collision free. For difficult parts, this is a much more economical way of growing the roadmap than going through all of the expansion, searching, and pruning steps.

VII. Experimental results

Throughout its development, the planner has been tested on jobs for actual ship blocks. The jobs are computed by a CAD system called TAS that has been developed within OSS. The TAS system determines sensing and welding targets and assigns parameters for the welding process.

The planner has been tested on more than a hundred jobs with up to 8 weld lines each. The manipulators used for planning were the 6 *dof* Motoman HP6S and the 5 *dof* Hirobo WR-L80. For the majority of jobs, initial solution paths could be computed on a standard PC in less than 1 second per weld line. The extra degree of freedom of the Motoman manipulator allowed it to weld paths in a few cases where the Hirobo WR-L80 were forced to change posture. If given a little time to improve the initial solution, the planner in almost all cases finds a solution where no or only a single change of position of the gantry is done per weld line.

The simple randomized gantry positioning procedure (Section III) often needs only a handful of attempts to select a gantry position from which all welding targets can be reached. Fig. 5 shows the failure rate for the IK procedure for a target or group of targets for gantry positions sampled uniformly at random from the $\mathcal{W}_{G',1}$ or $\mathcal{W}_{G',2}$ region. Only gantry positions within a distance R of every target were considered. The figure is for a 40 cm long weld line welded by the Hirobo WR-L80. As in Fig. 3, the vertical lines show the positions of the sensing and welding targets. The jagged graph shows the failure rate for a single target as a function of the tool position. The horizontal line segments show the failure rates for groups of targets. Each segment corresponds to one of the groups of Fig. 2. The figure shows that reaching a group of targets is harder than reaching a single target, but



5: Failure rate for IK procedure calls for single targets and groups of targets.

6: Hirobo WR-L80 manipulator at work at Odense Steel Shipyard.

not much harder. The failure rate for reaching all targets is about 80% which is adequate, considering the low cost of the IK procedure calls. The ship blocks are designed to allow them to be welded; therefore avoiding collisions is often also easy.

The computed paths have been tested for the welding of a ship block at the Hirobo WR-L80 installation (Fig. 6) at OSS. It was found that keeping the gantry fixed often led the manipulator to work at too high joint velocities. The cost measure of the planner was therefore changed to give higher priority to the minimization of this.

Following the work described in this paper, a version of the planner was implemented at OSS. The OSS planner uses the technique of a layered roadmap and grouping of targets, but differs in many of the details. Our planner expands the roadmap for all groups and repeats the expansion and shortest–path searching as long as time permits or until a path of sufficiently high quality has been found. The OSS planner on the other hand has a prioritized list of sets of groups to use for a path. The set of highest priority contains only the outermost group containing the entire job. If no path can be found for this set of groups, the planner progresses to the next set on its list. The OSS planner checks nodes for collisions before they are inserted in the roadmap and defers collision checking only for the edges. To have greater control of the type of movements generated, the OSS planner is not randomized. Instead the OSS planner has a small database of manually selected acceptable manipulator configurations. The OSS planner by a deterministic procedure selects gantry positions that yield manipulator configurations that are close to those of the database. If a ship block is found to be too difficult to weld, a few select configurations can be added to the database.

The OSS version of the planner (together with TAS) is in production use today at the shipyards B4 line of Motoman HP6S manipulators. The planner replaces a motion selection system based on parameterized robot programs that were recorded with a teach pendant. Goals for the new robot programming system were to find paths for a greater percentage of the weld lines and to be more flexible with respect to new robot tasks and robot installations.

VIII. Conclusion

State-of-the-art motion planning techniques have been applied for a welding application for gantry mounted manipulators at Odense Steel Shipyard. Results of this project has the shipyard to apply similar techniques in production for their new off-line robot programming system.

Of more general interest, it is shown how a base (or gantry) positioning algorithm can be incorporated in a roadmap planner. A simple random sampling based scheme was found to be adequate for tasks such as line welding. To control the growth of the roadmap, aggressive pruning of the roadmap was done. The pruning was guided by an application specific heuristic for what nodes are likely to be on a solution path.

Acknowledgment

The authors would like to thank Charlotte Jul Jacobsen for preparing jobs with TAS and Christian Lund Nielsen for details on the OSS implementation of the planner.

References

[1] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion*. MIT Press, 2005.

[2] G. Song, S. Miller, and N. M. Amato, "Customizing PRM roadmaps at query time," in *Proceedings IEEE International Conference on Robotics and Automation*, May 2001, pp. 1500–1505.

[3] R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," in *Proceedings IEEE International Conference on Robotics and Automation*, April 2000, pp. 521–528.

[4] C. L. Nielsen and L. E. Kavraki, "A two-level fuzzy PRM for manipulation planning," in *Proceedings of The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE Press, November 2000, pp. 1716–1722.

[5] L. Han and N. M. Amato, "A kinematics-based probabilistic roadmap method for closed chain systems," in *Proceedings Workshop on Algorithmic Foundations of Robotics*, 2000, pp. 233–246.

[6] H. Seraji, "Reachability analysis for base placement in mobile manipulators," *Journal of Robotic Systems*, vol. 12, no. 1, pp. 29–43, 1995.

[7] J. Cortés, "Motion planning algorithms for general closed-chain mechanisms," Ph.D. dissertation, Institut National Polytechnique de Toulouse, 2003.

[8] F. Ouezdou and S. Regnier, "General method for kinematic synthesis of manipulators with task specifications," *Robotica*, vol. 15, no. 6, pp. 653–661, November 1997.

[9] F. C. Park and J. E. Bobrow, "Efficient geometric algorithms for robot kinematic design," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 2, May 1995, pp. 2132–2137.

[10] K. Abdel-Malek and W. Yu, "On the placement of serial manipulators," in *Proceedings of the ASME 26th Biennial Mechanisms Conference*, 2000.

[11] G. Sanchez and J.-C. Latombe, "On delaying collision checking in PRM planning – application to multi-robot coordination," *International Journal of Robotics Research*, vol. 21, no. 1, pp. 5–26, January 2002.