Folding Cartons with Fixtures: A Motion Planning Approach

Liang Lu and Srinivas Akella, *Member, IEEE*

***Abstract*—Packaging products such as telephones and two-way radios after assembly is a common manufacturing task. Carton folding is a packaging operation typically performed by human operators or with fixed automation. We present a flexible method to fold cardboard cartons using fixtures; a carton blank is folded by moving it through a fixture with a robot. This method uses interchangeable fixtures to enable rapid changeovers between product models. We outline an approach to design a fixture given a carton and a folding sequence. We present an implemented motion planning algorithm that generates all folding sequences for a carton by modeling it kinematically as a many degree-of-freedom robot manipulator with revolute joints and branching links. Folding fixtures constrain the carton motion to paths consisting of line segments in its configuration space. We characterize the set of valid paths for these carton robots and generate them using the motion planner. To illustrate the method, we selected a folding sequence for an example carton, designed a fixture, and demonstrated folding of the carton from blanks with an industrial robot.**

***Index Terms*—Carton folding, flexible manufacturing, packaging, robot motion planning.**

I. Introduction

PACKAGING products after assembly is a common manufacturing task. Cartons folded from flat cardboard blanks are used to package products such as telephones, two-way radios, and automotive components. Carton folding operations occur in packaging, distribution, and warehousing centers, and are typically performed by human operators or with fixed automation. Carton folding by humans requires dextrous manipulation of the carton and can lead to repetitive stress injuries. Commercial carton folding machines are not designed to handle the different carton styles necessitated by product and line changes.

We present a method to fold cartons using fixtures, where a carton blank is folded by moving it through the fixture with a robot. The fixtures are interchangeable and enable rapid changeovers between product models. Folding a carton with a fixture requires generating valid folding sequences for the carton, designing a fixture for a selected folding sequence, and implementing the folding operations on the fixture with a robot. A carton is an articulated structure and we model it kinematically as a robot manipulator with revolute joints and branching links. We present a motion planning algorithm to automatically generate all folding sequences for a class of cardboard cartons folded from flat carton blanks with self-locking tabs (Fig. 1). We outline an approach to design a fixture given a carton and a folding sequence. Folding fixtures constrain the carton motion to paths consisting of line segments in its configuration space. We characterize the set of valid paths for these carton robots with many degrees of freedom, and generate feasible carton folding sequences with the motion planner. To illustrate the method, we selected a planner generated folding sequence for an example carton, designed a fixture using the design procedure, and demonstrated folding of the carton from blanks with an AdeptOne robot.

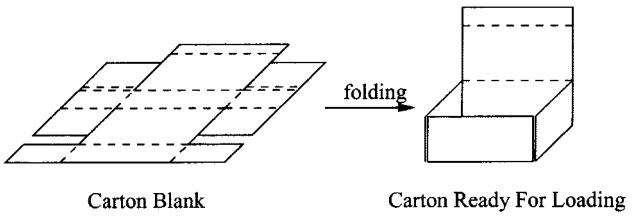


Fig. 1. The radio carton blank in its initial configuration and its folded configuration.

Automation of the carton folding process involves several subtasks including automatic folding sequence generation, automated folding fixture design, motion planning of the actuating robot, design of carton shapes for foldability, and efficient layout of carton blanks for cutting from stock. In this paper, we address the first and necessary task: generating physically valid folding sequences for consideration during generation of fixtures and motions of the actuating robot. This work is a first step toward our long term goal of identifying fixture-foldable sequences and automatically designing carton folding fixtures. This future work will require taking into account the feasible motions of the actuating robot, holding tool constraints, and the stiffnesses of the fold creases.

Carton folding is a task that involves the manipulation of articulated structures. A folding carton typically has many more degrees of freedom than the robot used to fold it. Our approach is to use the motion planner to aid the design of minimal complexity hardware by a human designer. The planner can potentially be used for similar shape modifying manufacturing tasks, such as sheet metal bending or in the design of 3-D microelectromechanical structures (MEMS) from hinged 2-D elements.

Manuscript received January 11, 1999; revised October 31, 1999. This paper was recommended for publication by Associate Editor L. Kavraki and Editor P. Luh upon evaluation of the reviewers' comments. This work was supported in part by the Beckman Institute and by Motorola through the Manufacturing Research Center at the University of Illinois at Urbana-Champaign. This paper was presented in part at the IEEE International Conference on Robotics and Automation, Detroit, MI, 1999.

L. Lu was with the Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA. He is now with the Cellular Subscriber Sector, Motorola, Inc., Libertyville, IL 60048 USA.

S. Akella was with the Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA. He is now with the Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: sakella@cs.rpi.edu).

Publisher Item Identifier S 1042-296X(00)06951-2.

After a review of related work in Section II, we outline the fixture design approach for a given carton and folding sequence in Section III. We then characterize the carton folding problem and present a motion planning algorithm to generate folding sequences in Section IV. In Section V, we illustrate the approach on different carton styles. We present a fixture designed for an example carton and a folding sequence implemented with a robot in Section VI. We conclude with a summary and outline future work in Section VII. Portions of this work appeared earlier in [1].

II. Related Work

The packaging encyclopedia [2] and the books by Hanlon [3] and Friedman and Kipnees [4] provide an introduction and overview of packaging. Carton folding machines for industrial use are typically available for cartons that are glued or taped in their final configuration. We are aware of at least one commercial machine to fold the class of cartons considered in this paper [5], but the machines cannot be easily adapted to different carton styles. The patent literature describes several carton folding designs equipped with rotating wheels [6], [7], spiral bars or spiral belts at different orientations [8], [9], or rotatable bar folders [10] that gradually fold the flaps as a carton blank passes through. Such devices with a large number of moving elements cannot be easily modified to handle other cartons.

The problem of carton folding shares geometric similarities with the problem of creating 3-D sheet metal parts from sheet metal blanks by bending. Both involve manipulating objects as their shape changes. However, sheet metal bending differs from carton folding in at least two aspects. First, a bent sheet metal part remains bent. Second, simultaneous sheet metal bends are usually performed only along collinear lines. Determination of bending sequences is a combinatorial problem coupled with the selection of bending tools. Most approaches focus on generating feasible solutions that may be suboptimal. de Vin *et al.* [11] describe a computer-aided process planning system to determine bending sequences for sheet-metal manufacturing. They generate reverse bending sequences for sheet-metal components with straight bending lines, bent on press brakes. The system uses rules to increase accuracy, minimize handling, select tools, and avoid collisions between parts and punch tools. Shpitalni and Saddan [12] describe a system to automatically generate bend sequences with up to 18 bend lines. They use domain-specific costs such as the number of tool changes and part reorientations as well as heuristics based on the number of tools used and lengths and locations of bends to guide the A* search algorithm. Radin *et al.* [13] present a two-stage algorithm that first rapidly generates a feasible bending sequence using collision avoidance heuristics and then searches for lower cost solutions without violating its time limitations. Inui *et al.* [14] developed an automatic planner for bending sheet metal parts. Gupta *et al.* [15] describe a fully automated process planning system to plan and execute bends with a robotic sheet metal bending press brake. Wang [16] develops methods to decompose sheet metal products into components for ease of manufacturing during cutting, bending, and assembly. In particular, he unfolds 3-D products into 2-D patterns, and identifies unfolding bend sequences that avoid collisions with tools. Wang and Bourne [17] use shape symmetry to reduce planning complexity for sheet metal layout planning, bend planning, part stacking, and assembly.

The geometric constraints in carton folding parallel those in assembly planning. Assembly planning by disassembly sequencing is a common technique. Mattikalli and Khosla [18] present a method to determine the constraints on motion of objects in an assembly from their contact geometry. Wilson and Latombe [19] developed the nondirectional blocking graph to efficiently generate monotone two-handed assembly plans. Wilson [20] has considered the geometric accessibility constraints on assembly resulting from assembly tools. A current trend in robotics research is toward reduced complexity devices [21] that use simple, cheap, and robust actuators and sensors. Of particular interest is the work of Goldberg and Moradi [22] on assembly planning for assembly machines that perform a pipelined series of part rotations and vertical insertions that require only one or two degrees of freedom.

We model cartons as robot manipulators with many degrees of freedom to generate folding sequences using motion planning techniques. The book by Latombe [23] provides a comprehensive introduction to motion planning. Developing practical motion planners for manipulator arms with many degrees of freedom is an area of ongoing research. Barraquand and Latombe [24] developed a randomized path planner (RPP) that uses a combination of potential field techniques and randomized search techniques. An RPP can effectively generate motion plans for robots with a large number of degrees of freedom. Kavraki *et al.* [25] construct probabilistic roadmaps for path planning in high dimensional configuration spaces. Their method first develops a graph representation of the roadmap by sampling points in the configuration space during a learning phase. In the query phase, the roadmap is searched for paths connecting the start and goal configurations. Gupta and Guo [26] develop a sequential framework with backtracking for motion planning of robots with many degrees of freedom. They decompose the problem into a sequence of planning problems for each link, and backtrack when no solution is found for a particular link. Amato *et al.* [27] describe an obstacle-based probabilistic roadmap technique for motion planning in cluttered environments that samples points on or near configuration space obstacles. Lozano-Perez [28] developed a simple algorithm that uses an approximate cell decomposition to plan collision-free motions of serial robot manipulators. The algorithm is exponential in the number of degrees of freedom of the robot. He generates an efficient approximation of the free space, built up recursively from one-dimensional slices of the configuration space. We use his recursive tree data structure, whose leaves represent legal ranges of configurations for the robot.

Recent work in computational geometry relates polyhedra and the polygonal shapes to which they can be unfolded, and develops connections to the art of origami. Lubiw and O'Rourke [29] present a dynamic programming algorithm to determine if a given polygon can be folded to a convex polytope. Biedl *et al.* [30] study unfoldings of two classes of nonconvex orthogonal polyhedra. Demaine *et al.* [31] show that a square piece of

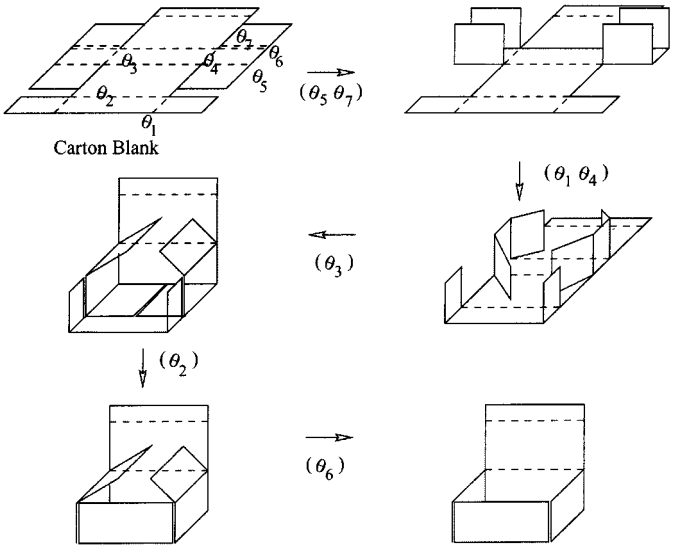


Fig. 2. An example folding sequence. Fold creases are indicated by dotted lines.

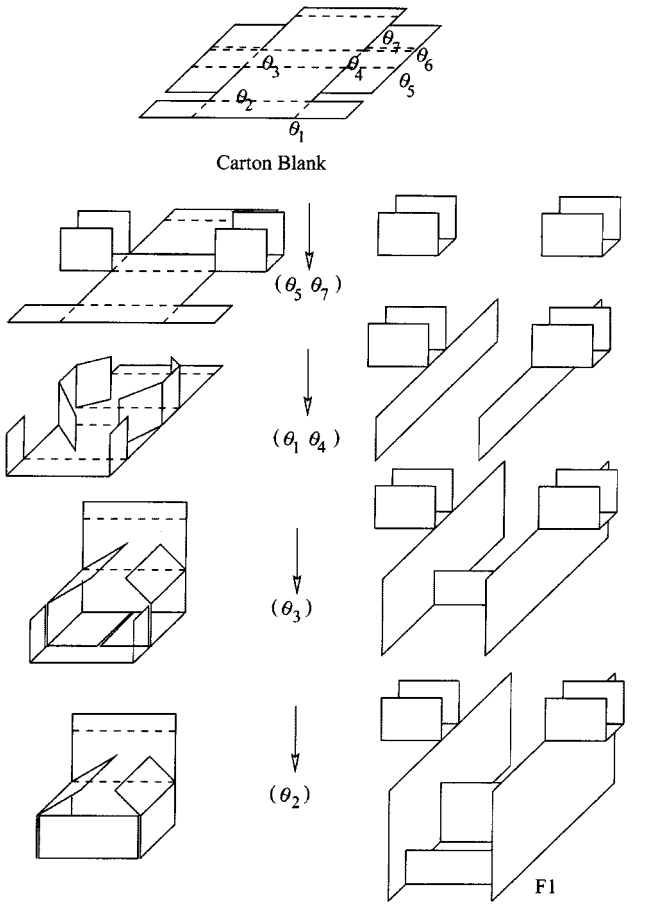


Fig. 3. Designing a folding fixture. Creating fixture elements to perform each fold of the folding sequence shown on the left for a downward carton translation results in the fixture F1 shown at bottom right. At each stage, fixture walls are created or modified to accomplish the corresponding fold. Note that unrestrained flaps can spring back at intermediate stages of the folding and that an additional fold is necessary to tuck in the upper flaps.

paper can be folded into any connected polygonal region, and further, any polyhedron can be folded from a single paper.

The geometric issues in carton folding are similar to those in creating 3-D MEMS from 2-D hinged elements. Pister *et al.* [32] describe a variety of 3-D structures created from 2-D hinged polysilicon structures made by surface micromachining. Syms [33] presents a process for self-assembly of 3-D microstructures that relies on rotation of 2-D surface micromachined parts due to surface tension forces from a meltable material. Yi and Liu [34] developed a method for magnetic out-of-plane actuation of hinged planar microstructures to create self-locking 3-D MEMS devices.

III. Folding with Fixtures

A carton blank is a flat cardboard cutout structure consisting of a set of panels, with fold creases separating adjacent connected panels (Fig. 1). Carton folding is often performed by humans using their hands both to fold the carton and to maintain the intermediate folded configurations of the carton panels.

A carton can usually be folded to its goal state in multiple ways that differ in the sequence of folds. See Fig. 2 for an example folding sequence. We fold carton blanks into cartons using fixtures. We design the fixture shape to match the carton for a selected folding sequence so that the carton is folded along its creases by being moved in contact with the fixture (Fig. 3). To fold the carton at a crease as it translates, we place a *wall* element perpendicular to the carton panel at the crease location. A wall is a flat plane whose length is chosen to be slightly greater than the crease length. The wall causes a carton panel to rotate along the crease and maintains the folded shape of the panels. For a chosen fold sequence, the distances between walls along the motion direction are selected to ensure the folds occur in the desired sequence. The walls are located an empirically determined distance away from the creases to ensure successful folds.

Multiple folds of the carton can be performed simultaneously by judicious design of the fixture. Making the fixture a passive structure and using an industrial robot to move the carton through the fixture leads to a robust design with a small number of moving elements. Since a folding sequence specifies only the relative orientations of the carton panels, a given folding sequence can have several instantiations in the world frame, each of which may yield a different folding fixture.

IV. Folding Sequence Generation

Identifying feasible folding sequences is central to the fixture design process. Our approach is to generate all valid folding sequences, independent of fixture shape, for a human designer to evaluate. We formulate the generation of folding sequences as a robot motion planning problem. We model a carton as an articulated robot with revolute joints and a branching sequence of links (Fig. 4). Since a carton has many degrees of freedom (12 in our examples), we exploit symmetry in the carton shape to reduce the number of joints to consider. The motions of panels on one side of the carton are assumed to mirror those on the other side. Each valid path from the initial robot configuration to the goal robot configuration determines a folding sequence. We seek to generate all possible paths, and to use one of them to design a folding fixture.

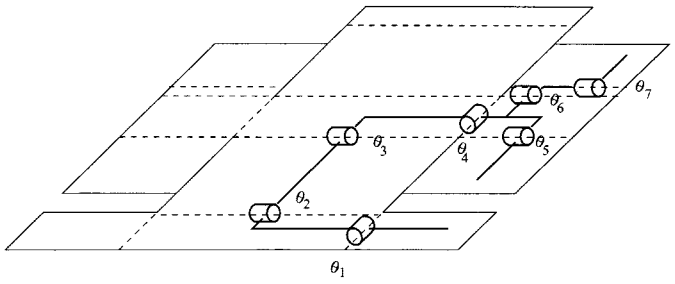


Fig. 4. The radio carton as an articulated manipulator robot with revolute joints and a branching sequence of links. Symmetry in the carton's shape makes it sufficient to model only one half of the carton as a robot.

The motion planning problem is stated as follows. *Given the carton shape and its kinematic representation as a robot, find all physically valid paths between its initial and goal configurations*. A physically valid path is one that satisfies motion constraints due to an abstract fixture and does not involve self-collisions of the carton panels. We identify valid paths independent of the fixture shape and feasible motions of the actuating robot, and do not model the actuating robot or the fixture as obstacles. From a motion planning perspective, the problem of generating folding sequences is challenging for the following reasons.

1) A carton robot has a large number of degrees of freedom, ranging from 7 to 9 for the example cartons, even after considering carton symmetry.
2) A carton robot is a nonserial manipulator with branching sequences of links.
3) To generate all folding sequences, we seek all possible paths, unlike most motion planning algorithms that look for a single path from the initial configuration to the goal configuration.

We can consider the joint motions to move a carton from its unfolded configuration to its folded configuration, or from its folded configuration to its unfolded configuration. Since a folded configuration has more constraints on the permissible joint motions than an unfolded configuration, it is more efficient to consider unfolding a carton from its folded configuration to its unfolded configuration. In what follows, we generate unfolding sequences rather than folding sequences.

A. Configuration Space Representation

The configuration of an object is a set of parameters that completely specify the position of every point of the object relative to a fixed reference frame. The *configuration space* $\mathcal{C}$ of an object [35] is the space of all configurations of the object, where each coordinate represents a degree of freedom. For example, the set of joint angles of an articulated robot define a configuration space, which is in general $(S^1)^n$ for n revolute joints. The configurations forbidden to the robot due to collisions with other objects or due to self-collisions constitute the set of configuration space obstacles $\mathcal{C}_{\text{obstacle}}$. The set of legal configurations of the robot constitute its free space $\mathcal{C}_{\text{free}}$.

The configuration space of a carton with n joints where the ith joint has a lower limit of low_i and an upper limit of $high_i$ is $[low_1, high_1]\times\cdots\times[low_i, high_i]\times\cdots\times[low_n, high_n]$. We use a recursive tree structure, developed by Lozano-Perez [28] for a serial manipulator, to represent the configuration space of the carton robot using one-dimensional slices of the configuration space. Each level of the tree contains a set of cells corresponding to discretized intervals of one joint of the robot. A leaf of the tree is a cell representing a range of robot configurations identified as belonging to $\mathcal{C}_{\text{free}}$ or $\mathcal{C}_{\text{obstacle}}$. We extend Lozano-Perez's tree representation to handle the branching links of the carton robot. We select a base panel and compute the tree representation for the longest chain of links as before. For each branching sequence of links from this main chain, we create additional levels of the tree. The levels for the first branching sequence are attached to the leaf nodes of the tree for the main chain. Each branching chain of links is represented in sequence, and the levels for the branching chains are concatenated. A leaf node in the resulting tree represents a cell in configuration space and encodes whether or not it is in free space.

We do not model the fixture or the actuating robot as obstacles, and assume there are no external obstacles that collide with the carton. Since a moving panel may collide with other panels as the carton is folded, the configuration space obstacles correspond to collisions between carton panels. When generating the configuration space tree, we check for collisions of each rotating panel with the stationary panels [36]. Our current implementation assumes all joints except the moving joint are fixed at the middle of their orientation ranges. An advantage of Lozano-Perez's tree representation is that variations in joint angles can be compensated for by swept volume computations. This gives a conservative representation of the free space so generated paths are guaranteed to be collision-free, although some valid paths may be pruned.

B. Motion Planning Formulation

The simplest approach to generating folding sequences is to assume only one joint can be moved at a time, and that each moving joint moves monotonically to its goal orientation. So we can test the $n!$ possible joint sequences for feasibility by checking at each stage of a sequence if the moving joint can get from its initial orientation to its goal orientation without any intermediate configurations being in $\mathcal{C}_{\text{obstacle}}$. However, this approach gives only a subset of the folding sequences since it cannot generate solutions where multiple panels move simultaneously.

To generate folding sequences where multiple joints may move together, we first characterize the paths the robot can take in its configuration space when folded by an abstract (uninstantiated) fixture. Each joint is unactuated and begins rotating when a panel it is attached to contacts a wall of the fixture. We model all joints as moving at the same constant angular velocity ω, and assume that once a joint starts moving, it rotates monotonically to completion. (Section IV-D discusses the case when joints rotate at different velocities.) Joints may begin moving simultaneously or with delays in motion. We look for paths that take a carton from its folded configuration to its unfolded configuration. A robot with n joints has $n!$ possible joint sequences. We represent the ith *joint sequence* by $(\theta_{i_1}, \theta_{i_2}, \theta_{i_3}, \cdots, \theta_{i_n})$ which means joint θ_{i_1} starts moving before or at the same time as θ_{i_2}, θ_{i_2} starts moving before or at the same time as θ_{i_3}, and so on. (Note that we use θ_{i_k} to represent joint i_k as well as its angular value.) For each such

joint sequence, we must also identify the rotational delay δ between successive joints. A delay δ_k means joint $\theta_{i_{k+1}}$ starts rotating after joint θ_{i_k} has rotated through an angle δ_k. So we represent a *delay sequence* corresponding to the joint sequence $(\theta_{i_1}, \theta_{i_2}, \theta_{i_3}, \cdots, \theta_{i_n})$ by $(\delta_1, \delta_2, \cdots, \delta_{n-1})$. A joint θ_{i_k} rotates from low_{i_k} to high_{i_k}. For a specified joint sequence and delay sequence, the motion of joint θ_{i_k} expressed in terms of the time t is given by

$$\begin{aligned} \theta_{i_k}(t) &= \omega(t - t_k^{\text{start}}) + \text{low}_{i_k}, \qquad t_k^{\text{start}} \le t \le t_k^{\text{end}} \\ t_k^{\text{start}} &= \sum_{j=1}^{k-1} \delta_j/\omega \\ t_k^{\text{end}} &= (\text{high}_{i_k} - \text{low}_{i_k})/\omega + \sum_{j=1}^{k-1} \delta_j/\omega. \end{aligned} \tag{1}$$

The dependence of the joint angles on each other, for a specified joint sequence and delay sequence, can be equivalently expressed by the following *motion constraints:*

$$\begin{aligned} \theta_{i_k}(t) = \theta_{i_{k-1}}(t) - \delta_{k-1} - \text{low}_{i_{k-1}} + \text{low}_{i_k}, \\ t_k^{\text{start}} \le t \le t_{k-1}^{\text{end}};\ k = 2, \cdots, n. \end{aligned} \tag{2}$$

The *initial configuration constraints* are

$$\theta_{i_k}(t) = \text{low}_{i_k}, \qquad t < t_k^{\text{start}};\ k = 1, \cdots, n. \tag{3}$$

The *goal configuration constraints* are

$$\theta_{i_k}(t) = \text{high}_{i_k}, \qquad t > t_k^{\text{end}};\ k = 1, \cdots, n. \tag{4}$$

The delay δ_k can assume any value in the interval $[0, \text{high}_{i_k} - \text{low}_{i_k}]$. If δ_k is zero, the two joints θ_{i_k} and $\theta_{i_{k+1}}$ begin moving simultaneously, and if δ_k is $(\text{high}_{i_k} - \text{low}_{i_k})$, joint $\theta_{i_{k+1}}$ begins moving only after joint θ_{i_k} has completed its motion. For the cartons we studied, $\text{low}_{i_k} = 90°$ and $\text{high}_{i_k} = 180°$ for all k. The configuration space of these carton robots is therefore $[90, 180]^n$, and $\delta_1, \delta_2, \cdots, \delta_{n-1}$ can assume any value from 0° to 90°.

When m of the n joints are in motion, there are $m-1$ active motion constraints and $n-m$ active initial or goal configuration constraints. Each constraint defines a constraint plane in the n-dimensional configuration space of the robot, and the $n-1$ total active constraint planes intersect along a line. So a path for the robot consists of a sequence of line segments. The maximum number of line segments in a path is $2n-1$. See Fig. 5 for an example path illustrating the 3-D case.

C. Search Algorithm

Our goal is to identify collision-free paths for the carton robot. A path or a folding sequence is defined by a joint sequence and a corresponding delay sequence to take the robot from its folded configuration to its unfolded configuration. A robot with n joints has $n!$ joint sequences. To identify valid δ values for each feasible joint sequence, we discretize each δ range into s intervals sampled at their midpoints. So each joint sequence has s^{n-1} possible delay sequences. We must identify the valid folding sequences among the $n!s^{n-1}$ sequences. The most straightforward algorithm would be to generate all $n!$ joint sequences, and for each joint sequence, generate the paths corresponding to each of the s^{n-1} possible delay sequences. For each path, if none of its line segments collides with a configuration space obstacle, the folding sequence is feasible.

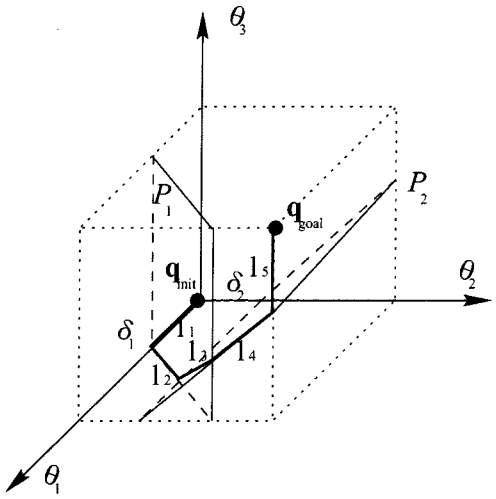


Fig. 5. Constraint planes and line segments along a path in a 3-D configuration space with $\mathbf{q}_{\text{init}}$ = (90, 90, 90) and $\mathbf{q}_{\text{goal}}$ = (180, 180, 180). The path τ composed of line segments l_1, l_2, l_3, l_4, and l_5 corresponds to joint sequence $(\theta_1, \theta_2, \theta_3)$ and delay sequence (δ_1, δ_2). The planes P_1 and P_2 correspond to the constraints $\theta_2 = \theta_1 - \delta_1$ and $\theta_3 = \theta_2 - \delta_2$ respectively. Line segment l_1 lies at the intersection of planes $\theta_2 = 90$ and $\theta_3 = 90$, and has length δ_1. Line segment l_2 lies at the intersection of P_1 with plane $\theta_3 = 90$, and the length of its projection on the θ_2 axis is δ_2. l_3 lies at the intersection of P_1 and P_2, l_4 lies at the intersection of P_2 with $\theta_1 = 180$, and l_5 lies at the intersection of $\theta_2 = 180$ with $\theta_1 = 180$.

We modify the above algorithm to more efficiently prune infeasible paths. For a given joint sequence, multiple paths to the goal may share one or more line segments. If a line segment collides with an obstacle, all the paths it belongs to are infeasible. Each partially instantiated delay sequence specifies a set of line segments, and we sequentially loop over them. For each instantiated δ_i, $1 \le i \le n-2$, test the defined line segment for a collision. If it does collide, the entire set of paths emanating from the colliding line segment can be classified as infeasible without instantiating the rest of the delay sequence. For each instantiation $\delta_1, \cdots, \delta_{n-2}$ of the delay sequence that yields a sequence of collision-free line segments, we find valid values for δ_{n-1} by exploiting spatial coherence of the obstacles. If the line segment defined for some value of δ_{n-1} collides with an obstacle, test the cells at the incremented value of δ_{n-1} that neighbor the collision cell. If those are collision-free, test the new line segment defined by the incremented value of δ_{n-1} for feasibility. Else again increment δ_{n-1} and repeat the process. See Fig. 6 for an example folding sequence generated by the algorithm. Note that the discretized representation of the configuration space and the delay values results in possible incompleteness of the planning method. That is, some valid solutions may not be returned by the planner.

The worst-case time complexity of the algorithm is exponential in the number of joints. By identifying the set of joints that can move first from the initial folded configuration, we eliminate some infeasible combinations and reduce the search time. We test if a joint can move first by checking if it has a

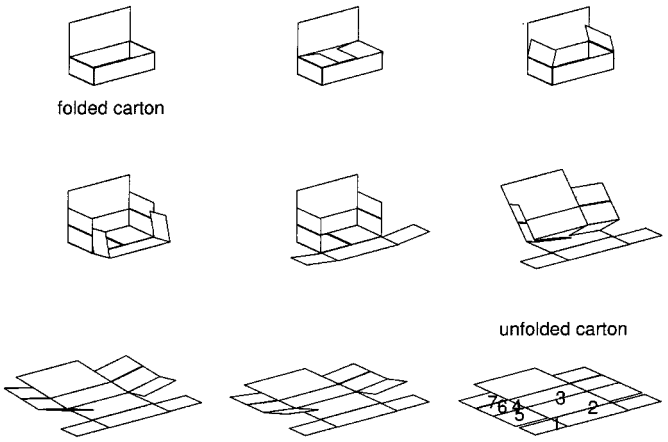


Fig. 6. Example (un)folding sequence for the radio carton generated by the algorithm has joint sequence (θ_6, θ_7, θ_2, θ_1, θ_3, θ_4, θ_5) and delay sequence (90, 90, 90, 90, 0, 10). The sequence goes from left to right and top to bottom. The bottom panel has a constant orientation during folding.

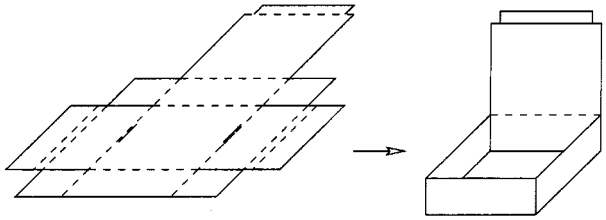


Fig. 7. The HP carton.

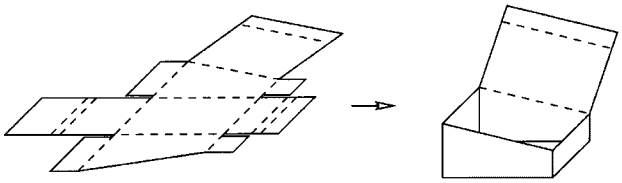


Fig. 8. The Slope carton.

collision-free configuration when rotated by a small amount, keeping all other joints at their start orientations. For each of the joints that can move first, we identify the set of joints that can potentially move second. Rapidly identifying the first two joints that can move means we explore only a subset of the $n!$ joint sequences.

D. Multiple Angular Velocity Formulation

Since all joints may not rotate at the same angular velocity, we can permit each joint θ_{i_k} to rotate with a constant angular velocity $\gamma_{i_k}\omega$, $\gamma_{i_k} \geq 1$. For a specified joint sequence and delay sequence, the motion of joint θ_{i_k} is now given by

$$\theta_{i_k}(t) = \gamma_{i_k}\omega(t - t_k^{\text{start}}) + \text{low}_{i_k}, \qquad t_k^{\text{start}} \leq t \leq t_k^{\text{end}}$$
$$t_k^{\text{start}} = \sum_{j=1}^{k-1} \delta_j/\gamma_{i_j}\omega$$
$$t_k^{\text{end}} = (\text{high}_{i_k} - \text{low}_{i_k})/\gamma_{i_k}\omega + t_k^{\text{start}}. \tag{5}$$

The initial and goal configuration constraints remain unchanged. The search algorithm is similar to that for equal angular velocities, with the additional consideration that for each joint sequence we must consider all possible valid angular velocity instantiations. We discretize the set of allowed angular velocities for each joint. When p joints can each have r angular velocities, the worst-case running time increases by a factor of $O(r^p)$.

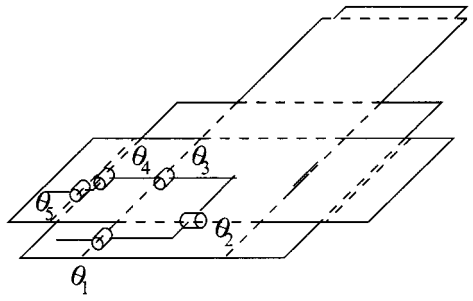


Fig. 9. Exploiting symmetry of the HP carton reduces the number of joints to model to 5.

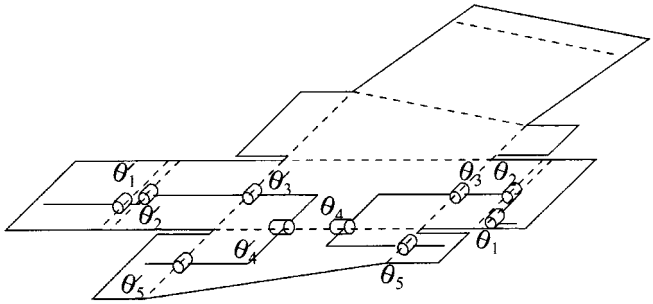


Fig. 10. The symmetric structure of the Slope carton can be exploited to model it as two robots with mirrored joint motions, reducing the number of joints to consider to 5.

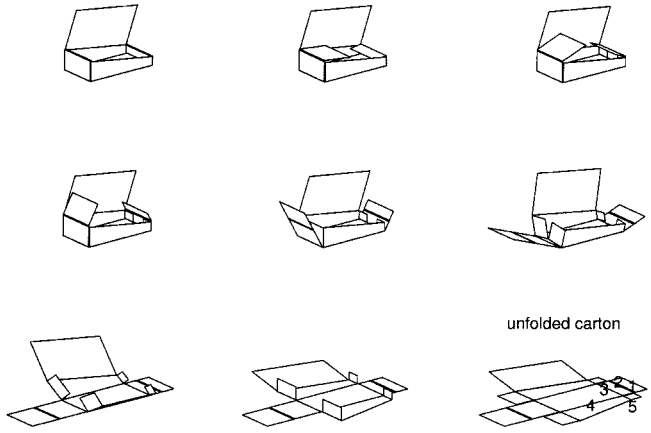


Fig. 11. An (un)folding sequence for the Slope carton, modeled as two five-joint robots, with joint sequence (θ_2, θ_1, θ_3, θ_4, θ_5) and delay sequence (90, 90, 50, 90).

E. Evaluating the Folding Sequences

The motion planner returns a potentially large number of valid folding sequences. The human designer must select a sequence based on its foldability by a fixture. Folding sequence generation can be made interactive by having the designer specify joint sequences to explore and folding sequences to discard. The designer must also consider the permitted motions

TABLE I

A COMPARISON OF THE RADIO, HP, AND SLOPE CARTON STYLES AND THE SYMMETRY-REDUCED HP AND SLOPE CARTONS. THE NUMBER OF DISCRETIZED JOINT INTERVALS AND s, THE NUMBER OF SAMPLING INTERVALS FOR EACH δ, ARE 10 FOR ALL CARTONS EXCEPT THE 9-JOINT SLOPE CARTON FOR WHICH THEY ARE 6, WITH CORRESPONDING SAMPLING RESOLUTIONS OF 10 DEGREES AND 18 DEGREES, RESPECTIVELY. (*) THE PATH GENERATION TIME FOR THE SLOPE CARTON IS THE TIME TO GENERATE A SINGLE DELAY SEQUENCE FOR EACH VALID JOINT SEQUENCE

Carton Style	No. of joints	Time to compute tree	Time to generate paths	No. of valid joint sequences	No. of tested joint sequences
Radio	7	2522 secs	877 secs	26	240
HP	7	2159 secs	1821 secs	352	480
Slope	9	6167 secs	1049 secs*	16,908	80,640
HP (symm.)	5	20.4 secs	4.8 secs	20	120
Slope (symm.)	5	66.2 secs	6.1 secs	22	120

TABLE II

A COMPARISON OF THE SYMMETRY-REDUCED CARTON STYLES WHEN MULTIPLE ANGULAR VELOCITIES ARE PERMITTED. EACH JOINT ROTATES WITH ANGULAR VELOCITY $\gamma_{i_k}\omega$, WHERE $\gamma_{i_k} \in \{1, 3\}$. THE NUMBER OF DISCRETIZED JOINT INTERVALS AND s, THE NUMBER OF SAMPLING INTERVALS FOR EACH δ, ARE 10 FOR ALL CARTONS WITH A CORRESPONDING SAMPLING RESOLUTION OF 10 DEGREES

Carton Style	No. of joints	Time to compute tree	Time to generate paths	No. of valid joint sequences	No. of tested joint sequences
Radio	7	2522 secs	120709 secs	26	240
HP (symm.)	5	20.4 secs	229.5 secs	22	120
Slope (symm.)	5	66.2 secs	309.8 secs	25	120

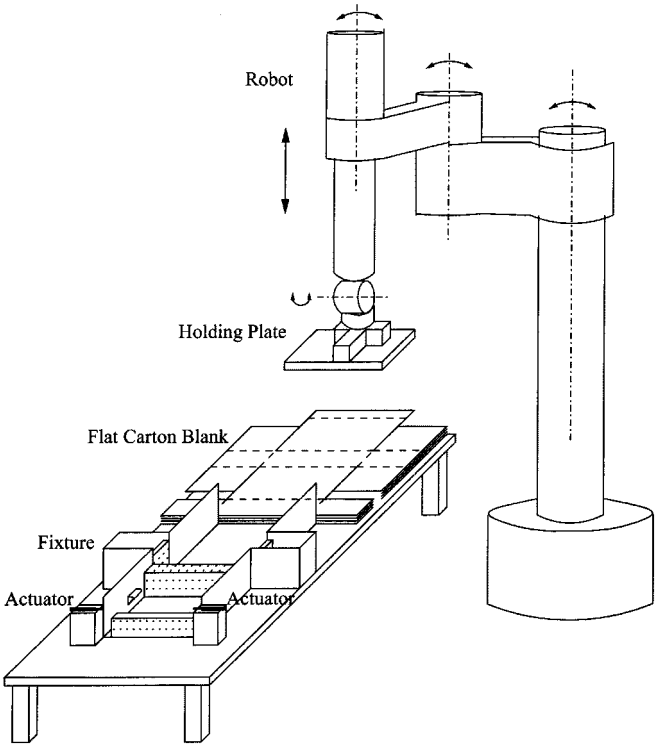


Fig. 12. Implemented fixture and AdeptOne robot.

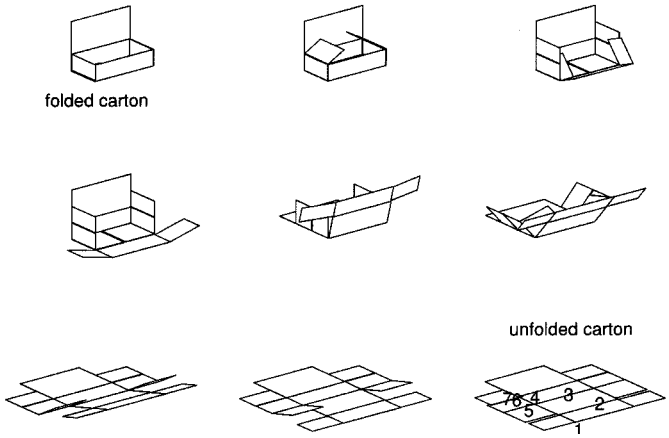


Fig. 13. The implemented folding sequence includes a rotation of the carton. The unfolding joint sequence is $(\theta_6, \theta_7, \theta_2, \theta_1, \theta_3, \theta_4, \theta_5)$, with delay sequence (0, 90, 90, 70, 20, 0) and the angular velocity of θ_4 thrice that of all other joints.

of the actuating robot when selecting a folding sequence for fixture design. The folding sequence instantiation in Fig. 6 requires only translation of the carton by the robot, while the implemented sequence in Fig. 13 involves a rotation as well.

Since a folding sequence specifies only the relative orientations of the carton panels, it may be instantiated in several ways. The instantiations depend on the feasible motions of the actuating robot, and whether it can only translate the carton or both rotate and translate it. Even when the carton can only be translated, any one of several panels may be selected to have a fixed orientation during folding. Thus, several different folding fixtures may be designed for a given folding sequence. The foldability of an instantiated sequence by a fixture is further determined by the holding tool constraints and the stiffnesses of the fold creases. Therefore, the foldability of the instantiations can differ significantly. In fact, not every physically valid folding sequence is guaranteed to have a valid fixture implementation.

V. CARTON STYLES

The motion planning algorithm can generate folding sequences for any carton style that can be modeled as a robot with revolute joints and branching sequences of links with no kinematic loops. Examples include the HP carton (Fig. 7) and the Slope carton (Fig. 8).

Since the generation of folding sequences is exponential in the number of joints, we seek to reduce the number of mod-

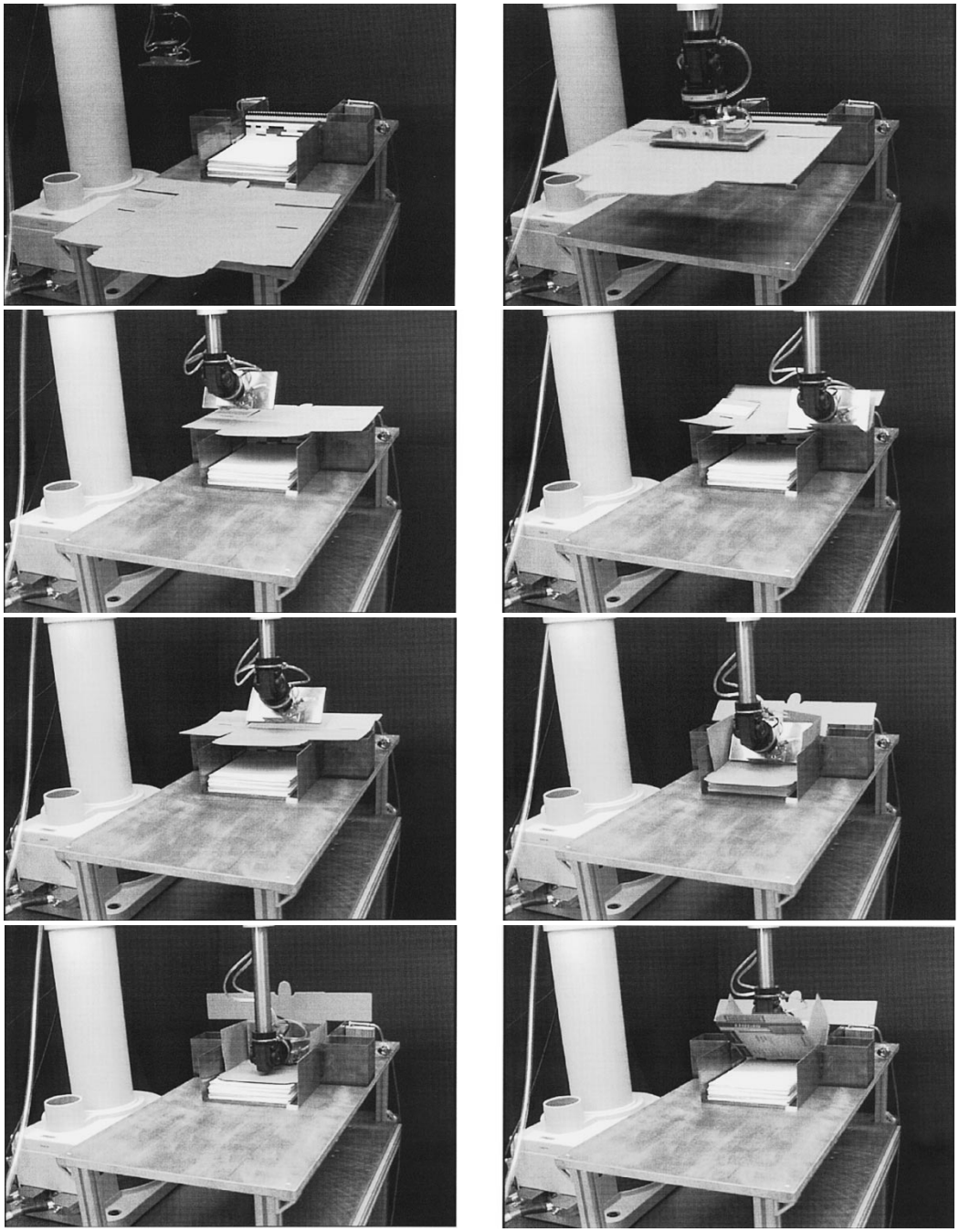


Fig. 14. The initial carton folding operations, shown from left to right, top to bottom. The carton blank is transferred to the fixture, where prefolding of critical creases is performed. A multiple-joint fold is then performed, followed by flap alignment operations and a rotation of the carton.

eled joints of the carton. We exploit the additional symmetry of the HP carton to model it as a serial manipulator (Fig. 9), reducing the number of robot joints from the original 12 to 5. The Slope carton can be modeled as two serial robots with different link dimensions and mirrored joint motions (Fig. 10), to reduce the number of joints from 12 to 5 for each robot. The configuration space trees for the individual robots are computed independently, and used jointly for path planning. The carton dimensions determine if these sequences must be tested for collisions of opposite panels. An example folding sequence for the Slope carton is shown in Fig. 11. Modeling a carton as a robot with fewer joints reduces the search time and the number of valid folding sequences, making human inspection easier. We assume the human specifies the symmetry-reduced joint representation as input. While existing algorithms can detect symmetry in polygonal and polyhedral shapes [17], they may not detect partial symmetry when the symmetric panels differ in their dimensions, as in the Slope carton.

VI. Implementation

We implemented the motion planner in C++ to generate folding sequences for a given carton robot. Table I compares the run times and feasible solutions for different carton styles when

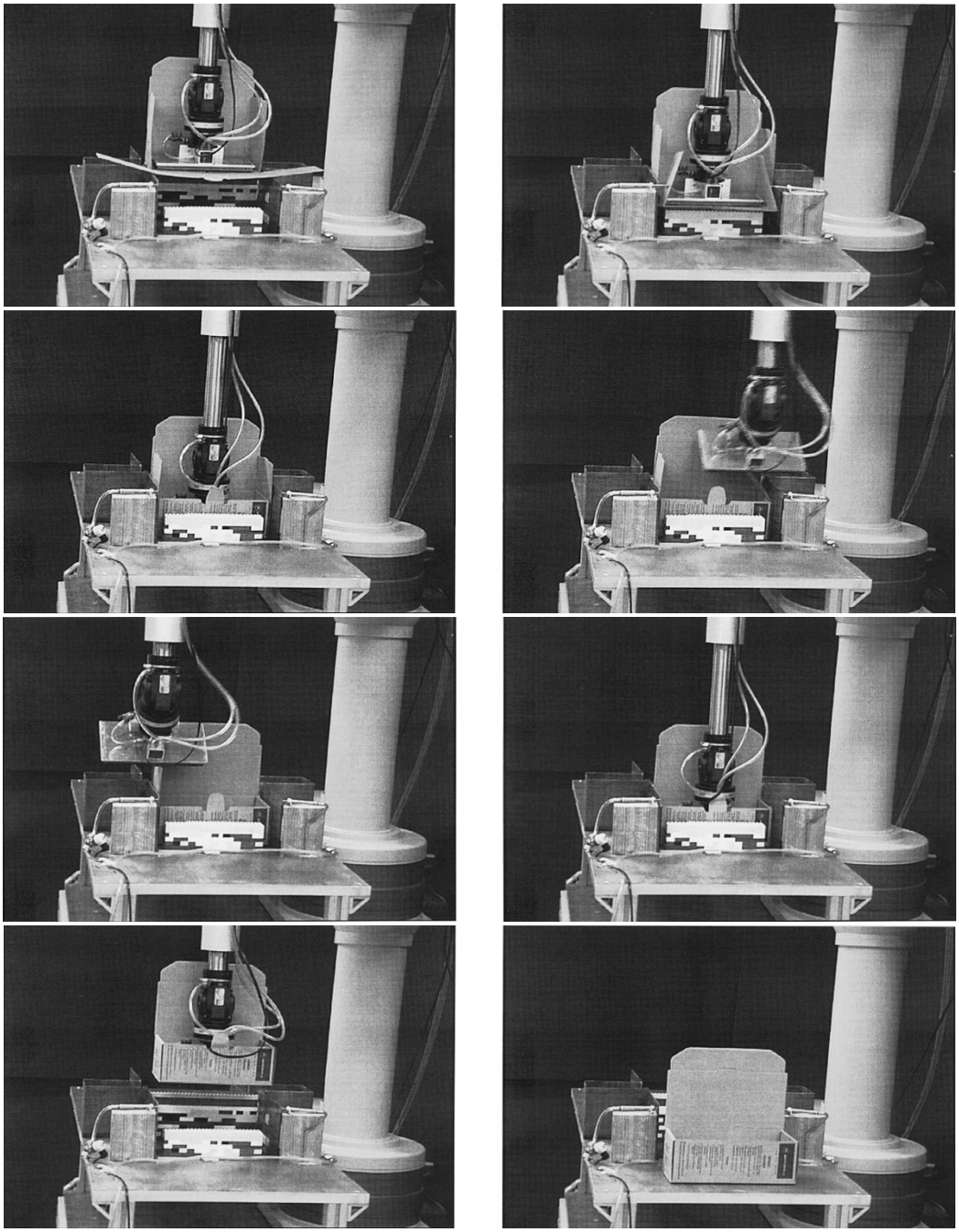


Fig. 15. Carton folding operations are continued in the front half of the fixture. The front panel is folded with the aid of two pneumatic actuators. The upper flaps are then folded and secured using self-locking tabs. The folded carton is removed from the fixture by the robot and placed for loading. The sequence runs from left to right and top to bottom.

all joints have the same angular velocity. All computations were on a 360-MHz Sun ULTRA 60. When computing the configuration space tree, our current implementation assumes all joints except the moving joint are fixed at the midpoints of their discretized intervals. The number of discretized joint intervals and the number of discretized delay intervals, s, are empirically chosen based on the number of joints and their motion ranges. Table II compares the symmetry-reduced cartons when each joint θ_{i_k} can rotate with an angular velocity $\gamma_{i_k}\omega$, where the candidate values for γ_{i_k} were empirically identified.

We demonstrated folding of the Radio carton style using an AdeptOne SCARA robot with an additional fifth joint (Fig. 12). The implemented folding sequence includes an intermediate rotation of the carton (Fig. 13). We designed a fixture for this sequence, following the approach of Section III, in an iterative process that considered the fixture shape and actuating robot motions. The fixture was constructed from sheet metal and Lego blocks to enable quick changes to the fixture during design and testing. The first set of folding operations occur in the back half of the fixture (Fig. 14) and the second set of folding operations occur in the front (Fig. 15), and are separated by the rotation of the carton. The fixture has two pneumatic actuators to aid the folding process. The robot uses a holding plate with vacuum suction cups to pick up the carton. Since the holding plate is di-

mensioned to enable the desired folds without collisions with the fixture, it may need to be changed along with the fixture when the carton style or size changes.

We used the same fixture and holding plate for experiments on two Radio carton models that differed slightly in their stiffness and crease locations. One model was successfully folded on 9 of 10 trials while the other was successfully folded on 10 of 16 trials. A trial was classified a failure if the carton had undesired folds, noticeable scuffing, an accidental collision of a panel and the holding plate occurred, or the self-locking tabs were not fully inserted. These difficulties can be eliminated with better fixture fabrication. The robot can fold a carton in about a minute in our current implementation [37]. Pipelining the folding operations or using multiple actuators in parallel can significantly reduce the execution time.

VII. Conclusion

We presented a flexible method to fold cardboard cartons using interchangeable fixtures; a carton blank is folded by moving it through a fixture with a robot. This paper makes the following contributions. First, it identifies the automation of carton folding by fixtures as a new task to which motion planning techniques can be applied. Second, it presents a motion planning formulation and an implemented algorithm that exploits the constraints on the motion of cartons folded using fixtures to automatically generate all folding sequences for a class of self-locking cartons folded from flat carton blanks. The planner generated sequences enable a human designer to visualize and identify fixture-foldable sequences and design the corresponding fixtures. Exploiting carton symmetry and careful selection of the panels to be modeled as a robot improve planner efficiency significantly. To illustrate this method, we selected a folding sequence for an example carton, designed a fixture, and demonstrated folding of the carton from blanks with an AdeptOne robot. This method enables rapid changeovers between different carton styles as it requires only a change in the fixture, the holding plate, and the robot program.

Our motion planning algorithm was developed to generate all possible folding sequences by exploiting the motion constraints due to fixtures. Cartons are modeled as robots with many degrees of freedom, and motion planners using probabilistic techniques are typically very effective in efficiently finding a path between the start and goal configurations for such robots. However, to the best of our knowledge, there has been no characterization of the performance of probabilistic planners in generating all possible paths between the start and the goal configurations.

The primary advantages of our approach are its simplicity and ease of implementation, and its ability to generate all solutions up to the discretization resolution. The primary disadvantage of the method is the exponential time required to generate all paths for cartons with a large number of degrees of freedom. However, many cartons can be reduced to robots with 5 to 7 degrees of freedom, enabling reasonable planner performance. The space requirements for the configuration space tree can be eliminated by the use instead of fast collision detection packages. Like other planners that use an approximate representation of the free space, it may not find some feasible solutions.

The motion planner can be used interactively by a human designer specifying folding sequences on which to focus. Future work to improve planner performance includes lazy evaluation of the configuration space obstacles, the use of fast collision detection packages (e.g., [38], [39]) to avoid explicitly computing the obstacles, characterizing the complexity of the free space of the carton robot, and parallelization of path generation. Identifying criteria to rank folding sequences and eliminate those that cannot be folded by a fixture will be useful. Several interesting problems remain to be addressed, including developing automatic planners to design fixtures, characterizing the effect of actuator robot degrees of freedom on the space of fixture designs, and designing cartons for efficient folding. This motion planning approach can potentially be applied to sheet metal bending and the design of 3-D MEMS structures from 2-D hinged elements [32].

Acknowledgment

The authors would like to thank J. Nguyen, S. Mok, and T. Babin at Motorola for their guidance and support. They would also like to thank S. Hutchinson for insightful feedback and helpful suggestions.

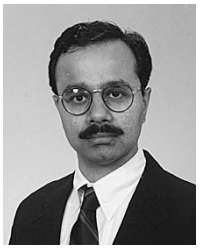
References

[1] L. Lu and S. Akella, "Folding cartons with fixtures: A motion planning approach," in *IEEE Int. Conf. on Robotics and Automation*, Detroit, MI, May 1999, pp. 1570–1576.
[2] A. L. Brody and K. S. Marsh, *Wiley Encyclopedia of Packaging Technology*, 2nd ed. New York: Wiley, 1997.
[3] J. F. Hanlon, *Package Engineering*, New York: McGraw-Hill, 1984.
[4] W. F. Friedman and J. J. Kipnees, *Industrial Packaging*. New York: Wiley, 1960.
[5] *Automatic tray and box former BEM-102*, Dorell Equipment, Inc., North Brunswick, NJ, 1999.
[6] C. R. Marschke, "Folding of paperboard sheets and the like," U.S. Patent 4 834 696, May 1989.
[7] J. McBride and R. Lile, "Corner laminating apparatus and method for cartons," U.S. Patent 4624653, Nov. 1986.
[8] B. Capdeboscq, "Sheet folding machine," U.S. Patent 4614512, July 1985.
[9] J. Takeda and H. Yoshizuka, "Folding device in a corrugated cardboard box making machine," U.S. Patent 5035683, Apr. 1990.
[10] H. D. Ward, "Box blank folding apparatus," U.S. Patent 4295841, Oct. 1981.
[11] L. J. de Vin, J. de Vries, A. H. Streppel, E. J. W. Klaassen, and H. J. J. Kals, "The generation of bending sequences in a CAPP system for sheet-metal components," *J. Mater. Processing Technol.*, vol. 41, pp. 331–339, 1994.
[12] M. Shpitalni and D. Saddan, "Automatic determination of bending sequence in sheet metal products," *Ann. CIRP*, vol. 43, no. 1, pp. 23–26, 1994.
[13] B. Radin, M. Shpitalni, and I. Hartman, "Two-stage algorithm for determination of the bending sequence in sheet metal products," *ASME J. Mechan. Design*, vol. 119, pp. 259–266, 1997.
[14] M. Inui, A. Kinosada, H. Suzuki, F. Kimura, and T. Sata, "Automatic process planning for sheet metal parts with bending simulation," in *ASME Winter Annu. Meeting, Symp. on Intelligent and Integrated Manufacturing Analysis and Synthesis*, 1987, pp. 245–258.
[15] S. K. Gupta, D. A. Bourne, K. H. Kim, and S. S. Krishnan, "Automated process planning for sheet metal bending operations," *J. Manufact. Syst.*, vol. 17, no. 5, pp. 338–360, 1998.
[16] C.-H. Wang, "Manufacturability-Driven Decomposition of Sheet Metal Products," Ph.D. dissertation, The Robotics Inst., Carnegie Mellon Univ., Pittsburgh, PA, Robotics Inst. Tech. Rep. CMU-RI-TR-97-35, Sept. 1997.

[17] C.-H. Wang and D. A. Bourne, "Using symmetry to reduce planning complexity: A case study in sheet-metal production," in *Proc. 1997 ASME Design Engineering Technical Conf.*, Sacramento, CA, Sept. 1997, DETC97DFM4327.
[18] R. S. Mattikalli and P. K. Khosla, "Motion constraints from contact geometry," in *IEEE Int. Conf. on Robotics and Automation*, Nice, France, May 1992, pp. 2178–2185.
[19] R. H. Wilson and J.-C. Latombe, "Geometric reasoning about mechanical assembly," *Artif. Intell.*, vol. 71, no. 2, pp. 371–396, Dec. 1994.
[20] R. Wilson, "A framework for geometric reasoning about tools in assembly," in *IEEE Int. Conf. on Robotics and Automation*, Minneapolis, MN, Apr. 1996, pp. 1837–1844.
[21] J. F. Canny and K. Y. Goldberg, "RISC for industrial robotics: Recent results and open problems," in *IEEE Int. Conf. on Robotics and Automation*, San Diego, CA, May 1994, pp. 1951–1958.
[22] K. Y. Goldberg and H. Moradi, "Compiling assembly plans into hard automation," in *IEEE Int. Conf. on Robotics and Automation*, Minneapolis, MN, Apr. 1996, pp. 1858–1863.
[23] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA: Kluwer, 1991.
[24] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.*, vol. 10, pp. 628–649, Dec. 1991.
[25] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, Aug. 1996.
[26] K. K. Gupta and Z. Guo, "Motion planning for many degrees of freedom: Sequential search with backtracking," *IEEE Trans. Robot. Automat.*, vol. 11, pp. 897–906, Dec. 1995.
[27] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *Robotics: The Algorithmic Perspective*, P. Agrawal, L. Kavraki, and M. T. Mason, Eds. Natick, MA: A. K. Peters, 1998.
[28] T. Lozano-Perez, "A simple motion-planning algorithm for general robot manipulators," *IEEE J. Robot. Automat.*, vol. RA-3, pp. 224–238, June 1987.
[29] A. Lubiw and J. O'Rourke, "When can a polygon fold to a polytope?," in *Proc. AMS Conf.*, Lawrenceville, NJ, Oct. 1996.
[30] T. Biedl, E. Demaine, M. Demaine, A. Lubiw, M. Overmars, J. O'Rourke, S. Robbins, and S. Whitesides, "Unfolding some classes of orthogonal polyhedra," in *Proc. 10th Canadian Conf. on Computational Geometry*, Aug. 1998, pp. 70–71.
[31] E. D. Demaine, M. L. Demaine, and J. S. B. Mitchell, "Folding flat silhouettes and wrapping polyhedral packages: New results in computational origami," in *Proc. 15th Annu. ACM Symp. Computational Geometry*, Miami Beach, FL, June 1999.
[32] K. S. J. Pister, M. W. Judy, S. R. Burgett, and R. S. Fearing, "Microfabricated hinges," *Sens. Actuators A, Phys.*, vol. 33, pp. 249–256, 1992.
[33] R. R. A. Syms, "Rotational self-assembly of complex microstructures by the surface tension of glass," *Sens. Actuators A, Phys.*, vol. 65, pp. 238–243, 1998.
[34] Y. Yi and C. Liu, "Parallel assembly of hinged microstructures using magnetic actuation," in *Proc. IEEE Solid-State Sensors and Actuators Workshop*, Hilton Head Island, SC, 1998, pp. 269–272.
[35] T. Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Trans. Comput.*, vol. C-32, pp. 108–120, Feb. 1983.
[36] L. Lu, "Folding cartons with fixtures: A motion planning approach," M.S. thesis, Depart. of Elect. Comput. Eng., Univ. of Illinois at Urbana-Champaign, May 1999.
[37] L. Lu and S. Akella, "Folding cartons with fixtures," in *Video Proc. IEEE Int. Conf. on Robotics and Automation*, Detroit, MI, May 1999.
[38] M. C. Lin, D. Manocha, J. Cohen, and S. Gottschalk, "Collision detection: Algorithms and applications," in *Algorithms for Robotic Motion and Manipulation*, J.-P. Laumond and M. Overmars, Eds. Wellesley, MA: A. K. Peters, 1997, pp. 129–142.
[39] B. Mirtich, "V-Clip: Fast and robust polyhedral collision detection," *ACM Trans. Graphics*, vol. 17, no. 3, pp. 177–208, July 1998.

Liang Lu received the B.Eng. degree in electrical engineering from Tsinghua University, Beijing, China, in 1990, the M.Eng. degree in electrical and computer engineering from the Chinese Academy of Sciences, Beijing, China, in 1993, and the M.S. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign in 1999.

From 1993 to 1996, he was with Hewlett Packard Medical Products, Qingdao, China. Currently, he is a Senior Software Engineer with Motorola, Inc., Libertyville, IL.



Srinivas Akella (S'90–M'96) received the B.Tech. degree in mechanical engineering from the Indian Institute of Technology, Madras, in 1989, and the Ph.D. degree in robotics from the School of Computer Science at Carnegie Mellon University, Pittsburgh, PA, in 1996.

He is currently an Assistant Professor in the Department of Computer Science at Rensselaer Polytechnic Institute, Troy, NY. From 1996 to 1999, he was a Beckman Fellow at the Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign. He received the National Talent Search Scholarship from the Government of India in 1984 and the J. N. Tata Scholarship in 1989. His research interests include robotic manipulation and motion planning, and developing planning and design algorithms for manufacturing tasks.