

Automated Feeding of Industrial Parts with Modular Blades: Design Software, Physical Experiments, and an Improved Algorithm

Onno C. Goemans, Marshall T. Anderson, Ken Goldberg and A. Frank van der Stappen

Abstract—The “blade”, a three-parameter primitive for part feeding, and a complete (quasi-static) algorithm for its design (which runs in $O(n^6)$ time) were introduced in [10]. Here, we present our implementation of the geometric blade design algorithm, a reconfigurable modular blade hardware design, and physical experiments: we report experiments with 14 blade configurations and 3 test parts. We then present an improved blade design algorithm which takes into account the part behaviors that are observed in the physical experiments, yet that are not considered by the quasi-static algorithm.

I. INTRODUCTION

A *part feeder* takes in a stream of identical parts in arbitrary orientations and outputs them in a single orientation. The oldest and still most common approach to automated feeding is the vibratory bowl feeder. It consists of a bowl, which is partially filled with (identical) parts, and a helical metal track that starts at the bottom and winds its way up along the bowl [5]. The bowl and track undergo an asymmetric helical vibration that causes the parts to move up the track, where they encounter a sequence of mechanical devices. Most of these devices are filters that serve to reject (force back to the bottom of the bowl) parts in all orientations except for the desired one. A stream of oriented parts emerges at the top after successfully running the gauntlet. A device or sequence of devices is said to *feed* a part if it allows only one specific orientation of the part to pass.

For the manufacturing of feeders, engineers still rely mostly on skill, experience, and ad-hoc guidelines. Currently, the largest cost factor of bowl feeder production is the design of the custom mechanisms tailored to feed a specific part. The expenses and time associated with the design of part feeders remains a barrier to flexible automation. In a typical scenario, the feeder bowl takes up to 200 hours to manufacture; 95 % of this time is spent designing the bowl [11]. It is evident that automation of the design process would greatly reduce the production cost and time.

To aid in the design of bowl feeder layouts, researchers have used simulation [2], [13], [15], [17], [18], heuristics [14], and genetic algorithms [7]. Geometric analysis tools have been developed that help designers visualize the configuration space of a given combination of part and bowl layout [6]. Research to single reorientation and rejection



Fig. 1. Reconfigurable modular hardware implementation of the blade device, mounted on a vibratory device. Three parts are on the track, moving from left to right, of which the rightmost is output. The “knocker” wiper (section IV) is the triangular shape attached to the side of the rightmost end of the track.

mechanisms has been focused at the design of traps, which are devices constructed by removing sections of the track [1], [3], [11]. A common characteristic of the existing solutions is that the resulting designs only apply to 2D parts.

In [10], a new class of geometric primitives was introduced that can feed a broad class of 3D parts by reorienting and rejecting all but a desired orientation. The design of these so called *blades* is inspired by similar devices used in existing bowl feeder systems. In addition, an algorithm to automate blade design was developed, which we will refer to as the *quasi-static (design) algorithm*; it is based on a quasi-static part motion model. This algorithm runs in $O(n^6)$ time and is *complete*; i.e., it identifies the set of all existing valid blade designs. We denote this set by S_{qs} .

In this paper, we present a series of new developments on blade design. We discuss a software implementation of the blade design algorithm as well as a modular hardware implementation of the blade device (Fig. 1). Employing both hard- and software, we conduct physical experiments based on design reports generated by the design software and we report our findings. In these experiments, we observe discrepancies between the part-blade interaction as predicted by the quasi-static algorithm and as observed in the physical experiments; we refer to these as the *observed discrepancies*. Finally, we propose an improved design algorithm, the *augmented (design) algorithm*, that expands the quasi-static algorithm by taking into account the observed discrepancies.

The *design software* we present implements the quasi-

Onno C. Goemans and A. Frank van der Stappen are with the Dept. of Information and Computing Sciences, Utrecht University, PO Box 80089, 3508 TB Utrecht, The Netherlands. *Email*: onno@cs.uu.nl, frankst@cs.uu.nl

Marshall T. Anderson is with the Dept. of IEOR, and Ken Goldberg is with the Dept. of IEOR, EECS and iSchool, University of California at Berkeley, Berkeley, CA 94720, USA. *Email*: mtanderson@berkeley.edu, goldberg@ieor.berkeley.edu

static design algorithm and is written in Mathematica. It takes 3D polyhedral surface model and center of mass position as input—these can be generated with standard CAD-software—and outputs S_{qs} . In addition, we present and implement heuristics to extract a number of blade designs to represent the continuous set of blade designs S_{qs} . The presented heuristics aim to report the designs that are least sensitive to tolerances in part motion and shape. We refer to these heuristics as *selection heuristics* and to the software output as the *design report*.

Furthermore, we present a reconfigurable modular *hardware implementation* of the blade device that allows us to evaluate the design reports and further study the blade device. To this end, we have conducted experiments with three parts. In order to acquire design reports for these physical parts, we apply CAD-software to subsequently construct detailed 3D models and compute their centers of mass and polyhedral approximations; finally, the latter two are input to the design software to generate the reports.

We report results on experiments with 14 blade configurations, which serve as a first exploration of the practical application of the blade. Although the aforementioned heuristics already introduce a certain measure of insensitivity to the observed discrepancies, these experiments suggest that these discrepancies must be (explicitly) taken into account to immediately produce practical blade designs. We discuss the effect of the observed discrepancies on the distinct phases of the part-blade interaction.

Lastly, we present the augmented algorithm, which can be outlined as follows. We start by executing a first phase of the quasi-static algorithm, resulting in an intermediate solution. Intuitively, this solution serves as a skeleton, which we then “pad” with a layer of uncertainty introduced by the observed discrepancies. The amount of padding, which varies and is tailored to each element of the skeleton, is approximated by external physics software—we treat this software as a black box. Finally, the second and last phase of the quasi-static algorithm executes. In short, the augmented algorithm extends the output S_{qs} of the quasi-static algorithm by incorporating the effect of the observed discrepancies on the feeding behavior of the blade.

This paper is organized as follows. In section II, we revisit the blade definition and modeling as presented in [10]. In section III, we present a discussion of our software implementation, details on the quasi-static design algorithm and the selection heuristics. In section IV, we describe the hardware implementation and address physical experiments. In section V, we further discuss the observed discrepancies and describe the augmented algorithm. We conclude in section VI with a brief summary, several remarks and points of future work.

II. MODELING THE BLADE

This section aims to establish an understanding of the blade concept as presented in previous work. For illustrations and a more detailed explanation, we refer to [10].

A. Basics

A blade is a metal plate attached to the feeder wall. This plate is parallel to the track and consists of a triangular and a rectangular shaped segment. A blade is characterized by three parameters: the blade angle, φ , expressing the slope of the triangular segment; the blade height, h , specifying the distance between track floor and blade; and the blade width, w , describing the width of rectangular shaped segment. Hence, a blade can be denoted by $B(\varphi, h, w)$.

Intuitively, the blade operates as follows. A part moves up the track toward the blade in an arbitrary stable pose. As soon as it reaches the blade, the part starts moving along the edge of the triangular segment and at some point breaks contact with the track wall. Simultaneously, the part changes its orientation as it moves from its stable pose against the track wall to a stable pose against the blade edge. The specific reorientation of the part depends on both the blade height and blade angle. From there, the part moves onto the rectangular segment, at which the rejection of parts takes place. The rejection depends on the blade width, as the blade width dictates whether center of mass C is supported by the track floor. Now, to realize that the blade feeds P , we select a width such that all but one part orientation is rejected. The role of the reorientation phase is to manipulate the stable track poses so that such a blade width exists.

In the following we discuss assumptions and modeling aspects; these apply to both the (implemented) quasi-static and augmented algorithm. We consider rigid three-dimensional polyhedral parts of a single type, all of which are assumed to be identical. As in the earlier works on mechanisms for vibratory feeders, we assume these parts move along a flat track without interfering with each other. We also assume the position of the center of mass C to be known, as it dictates the stability of the part. While moving up the track, a part rests on the track in a stable pose: the part rests on the track floor and against the track wall, where both contacts support C . The stable floor contact is the result of gravity, while a slight tilt of the feeder track assures a stable wall contact. Finally, the part is rejected, i.e. falls of the track, when its center of mass is no longer supported by the track floor. The radius of the helical track is assumed to be large compared to the dimensions of the part, thus allowing us to approximate the section of the track as linear.

Additionally, for the quasi-static algorithm, the part motion is assumed to be quasi-static and the friction between part and blade is assumed to be zero. In general, these last assumptions will be selectively relaxed in the augmented algorithm; the level of relaxation will depend both on the type of employed physics software and on the eventual implementation of the augmented algorithm.

Finally, the quasi-static algorithm automates the design of blades and is complete, i.e. it identifies all existing valid blade designs. As input it receives a polyhedral part P and center of mass C . The algorithm outputs the set S_{qs} of all valid blade designs that feed P .

B. Track Poses

While moving up the track, the part settles in a stable pose, thus discretizing its set of possible orientations. A stable track placement consists of a stable contact with (track) floor and with the (track) wall.

A stable *floor contact* is a placement in which the part rests on the floor with a convex hull face that supports C . Observe that such a contact discretizes two of the three degrees of rotational freedom. We denote P in a given floor contact by P^f , where f specifies the stable convex hull face on which P rests. The blade interaction is assumed not to change the floor contact of P ; we will revisit this assumption in section V on physical experiments.

The second aspect of a stable pose, the stable *wall contact*, is ensured by the earlier mentioned track tilt. This contact discretizes the remaining degree of freedom, the *roll orientation*, which is the rotation about the axis perpendicular to the track floor. Let P_{\perp}^f be the orthogonal projection of P^f on the floor. A stable wall contact for P^f corresponds to a roll orientation in which P_{\perp}^f rests with a stable edge against the track wall. Combing this with the above, we denote a (stable) track pose as P_e^f , where e specifies the stable edge of P_{\perp}^f . There exist $O(n^2)$ track poses .

C. Blade Poses

Let us start with two notes on the modeling of the blade itself. Firstly, we model the blade as a planar surface which is parallel to the track and defined by the earlier introduced parameters. Let us refer to the plane in which the blade lies as the *blade plane*. Secondly, the blade height h selects a cross-section of P^f with which P^f rests against the blade. This cross-section is the polygonal intersection of P^f and the blade plane, which we denote as $P^f(h)$. The location of the center of mass of $P^f(h)$ is the orthogonal projection of C onto the blade plane.

Sharing the concept of floor contact, the blade pose differs from the track pose in that the stable wall contact is replaced by a stable blade contact. Given a blade at height h , a stable blade contact corresponds to a roll orientation in which the convex hull of $P^f(h)$ rests with a stable edge against the blade. Let us denote a blade pose as P_{θ}^f , where θ specifies roll orientation with respect to a fixed world frame.

D. Part Reorientation

We start off with a general notion. A core concept is that we model reorientation and the subsequent rejection for each individual track pose P_e^f ; i.e., for each P_e^f , we model the effect of the blade parameters φ , h and w . The resulting set of models enables us to quickly determine the effect of any given $B = B(\varphi, h, w)$ on all track poses.

With this concept in mind, we discuss how the triangular segment of the blade, defined by φ and h , rotates a given track pose P_e^f to a blade pose P_{θ}^f . Recall that this reorientation process only affects the roll orientation of P_e^f . In the following discussion, we respectively address the role of h and φ .

The reorientation of P_e^f for a fixed $h = H$ effectively is the planar reorientation of $P^f(H)$, and visa versa. The choice of φ determines how the blade reorients polygon $P^f(H)$ from its known initial orientation to a stable blade pose [4]. Let us denote the allowed φ -interval as $[\varphi_l, \varphi_u]$. The convex hull of $P^f(H)$ features a set of stable edges that correspond to a set of stable roll orientations. Let $\Theta_h^f = \{\theta_1, \dots, \theta_k\}$ be the ordered sequence of stable roll orientations of $P^f(h)$. The general idea of the reorientation mechanism is as follows. The blades at $h = H$ can be grouped into sets of blades that reorient $P^f(H)$ to the same $\theta_i \in \Theta_h^f$, each of which corresponds to continuous subsets of $[\varphi_l, \varphi_u]$. Expressed in terms of track and blade poses, we summarize this concept: for a given P_e^f , the blades at h can be grouped into sets of blades that map P_e^f to the same P_{θ}^f , where each set corresponds to a continuous sub-interval of $[\varphi_l, \varphi_u]$.

E. Part Rejection

At this point, the blade has reoriented track pose P_e^f into a blade pose P_{θ}^f which then arrives at the rectangular segment of the blade. Recall that P_{θ}^f is rejected when the floor does not support the center of mass of P_{θ}^f ; the occurrence of such an event depends on the blade width. To capture this notion, we introduce the *critical blade width*, which is the maximum blade width that P_{θ}^f survives. We denote the critical width for a given φ and h by w_c .

III. DESIGN ALGORITHM & IMPLEMENTATION

The first subsection briefly reviews technical aspects of the software implementation of the quasi-static algorithm. In subsection III-B, we present an overview of the quasi-static algorithm itself [10], to which illustrations generated by the design software are added. In subsection III-C, we further examine S_{qs} —the set of valid blade designs generated by the design software—to develop insight into its structure. In the last subsection, we present the selection heuristics to extract the design report from S_{qs} .

A. Software Implementation

We have developed blade design software in Mathematica that implements the complete quasi-static design algorithm. The design software takes a 3D polyhedral surface model ('AutoCAD DXF' or 'STL' format) and C as input, both of which can be generated using standard CAD-software. Given this input, the software computes S_{qs} . In addition, we have implemented the selection heuristics, so as to enable the design software to construct a design report based on S_{qs} .

For 'part A'—the running example in this paper—and two other test parts, we constructed detailed 3D models and calculated their center of mass and polyhedral representations with CAD-software (Fig. 2). The polyhedral model of part A consists of approximately 250 polygons, which we believe to be a sufficiently accurate approximation of the real part. The design reports of these three parts are available on-line [9].

On a system with a AMD Athlon 3000+ CPU, 1.5GB RAM and Windows XP as operating system, the design software generates S_{qs} for this model of part A in about half



Fig. 2. From left to right: photo of part A, its CAD-model and polyhedral approximation.

a minute. However, no effort has been made to optimize the design software; we estimate that an optimized implementation would be able to generate a solution in a few seconds.

B. Design algorithm

After the identification of the track poses (subsection II-B), the design algorithm generates S_{qs} in two phases: firstly, it models the effect of the three blade parameters on each track pose P_e^f , resulting in set of separate models; secondly, it combines this set of models in one structure and extracts S_{qs} . The three-dimensional space in which both phases take place is spanned by the blade parameters: height h , blade angle φ and width w ; i.e., each point in this space describes a unique blade. We refer to this space as the *blade space*.

In the first of the aforementioned phases, the *modeling phase*, we forge the concepts from the previous section into a function in blade space. Such a function describes the impact of the blade parameters on the manipulation of a track pose. Let us consider an arbitrary track pose P_e^f . Firstly, the choice of φ and h determines the reorientation of P_e^f to some blade pose P_θ^f (section II-D). Secondly, once we have P_θ^f , we can calculate the corresponding critical blade width (section II-E). Concatenating these two steps, we can conclude that the choice of φ and h directly determines the critical blade width that P_e^f can survive.

We capture the above relation in a function that maps φ and h to the corresponding critical blade width w_c . This function defines a surface in the blade space, which we refer to as the *critical surface* (Fig. 4a). We note that the blade width w is specified as the distance between the outer edges of the rectangular blade segment and the track floor; a negative w value specifies a blade width that is larger than the width of the track floor. In conclusion, the critical surface has the following property: any point below the surface corresponds to a blade rejecting P_e^f , while any point above the surface corresponds to a blade that P_e^f survives.

Let us summarize the modeling phase. Each critical surface corresponds to *one* specific track pose and describes the impact of all possible blade parameterizations on this specific track pose. There exist $O(n^2)$ track poses, so the blade space contains $O(n^2)$ critical surfaces at the end of the modeling phase. These critical surfaces form an arrangement [8] of surfaces in the blade space (Fig. 4b) that allows us to identify for any given blade which track poses will survive.

We continue with the second phase of the quasi-static algorithm, the *extraction phase*, which operates on the

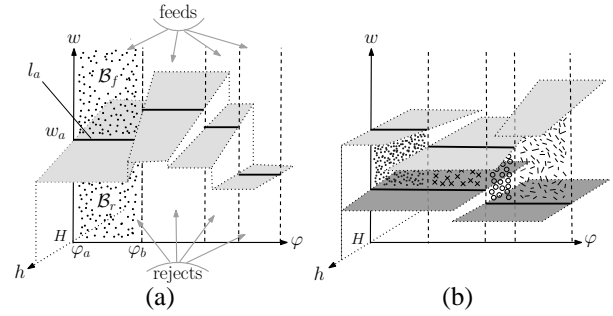


Fig. 3. Slices of (a) the critical surface and (b) first level with a highlighted cross-section at $h = H$. In (a) at $h = H$, there exist four φ -ranges, each of which reorients the track pose to a different blade poses. In (b), the cross-section at $h = H$ intersects four distinct (differently marked) cells.

arrangement of critical surfaces. Recall that we want to reject all but one track pose. In terms of our model, this objective translates to computing the set of points in the blade space that are below all but one critical surfaces. Such a set is referred to as the *first level* [8] of an arrangement of (critical) surfaces, and in our case forms the *valid solution set* S_{qs} (Fig. 4c).

C. First Level

We further examine the critical surface, critical arrangement and first level, and discuss several of their properties.

A critical surface is comprised of a set of surface patches which subdivide the blade space in a number of 3D cells; each cell consists blade designs of equal *feeding behavior*. Let us consider a cross-section of the critical surface of a given track pose P_e^f at $h = H$, as depicted in Fig. 3a. This 2D space contains a number of cross-sections of the aforementioned surface patches, which form horizontal line segments. Furthermore, let us consider segment l_a in Fig. 3a. For all $B(\varphi, H)$ with $\varphi \in [\varphi_a, \varphi_b]$ reorient P_e^f to one specific blade pose, P_a^f , where $[\varphi_a, \varphi_b]$ is the blade angle interval occupied by l_a (subsection II-D). Also, l_a splits the w -range in two at critical width w_a ; for all φ in $[\varphi_a, \varphi_b]$, blade designs $B(\varphi, H, w)$ with $w \geq w_a$ and $w < w_a$ respectively feed and reject P_a^f (subsection II-E). Intuitively, l_a thus defines a pair of rectangular 2D subspaces, B_f and B_r , each of which specifies a set of blade designs of equal *feeding behavior*. Generalizing this idea over all surface patch cross-sections and all h , the critical surface defines a discrete set of 3D cells, each of which consists of blade designs of equal feeding behavior.

With each critical surface subdividing the blade space into cells, the arrangement of critical surfaces also subdivides the blade space: an arrangement cell is defined as the blade space that intersects exactly one cell of each of the $O(n^2)$ critical surfaces. Consequently, each arrangement cell specifies a set of blade designs of equal feeding behavior for any and all track poses. We say that an arrangement cell satisfies the feeding property when the comprising blade designs feed the part.

With the above in mind, the first level consists of the set of arrangement cells that satisfy the feeding property (Fig. 3b). In summary, the first level S_{qs} consists of a number of

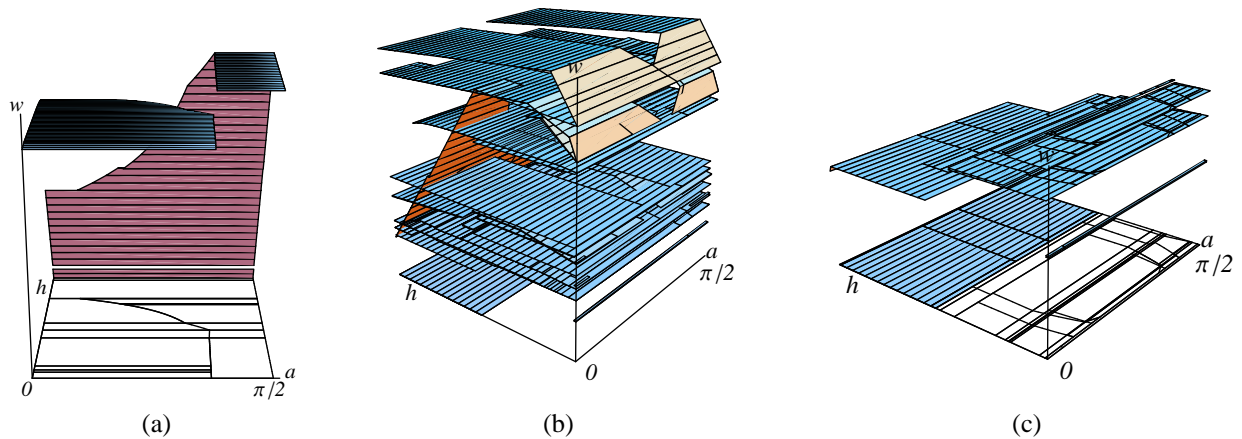


Fig. 4. Polygonal approximations of computed structures in the blade space (a -axis specifies φ) for part A, generated by the design software: (a) an example of a critical surface, including its planar projection, (b) the arrangement of critical surfaces, (c) the first level, which defines S_{qs}

three-dimensional cells, each of which is comprised of blade designs with equal feeding behavior.

D. Selection Heuristics & Design Report

In this section, we discuss the selection heuristics and their application to create a design report representing S , which can be either S_{qs} or S_{dy} . Important to note is that their application is a post-processing step that executes after the (quasi-static or augmented) design algorithm; the resulting set of blades is no longer complete. We can generally distinguish three categories of selection heuristics: firstly, *subdivision heuristics* subdivide S into a finite number of subsets, each of which is to be represented with one blade design; secondly, *representation heuristics* determine which blade design is to represent a given subset; and thirdly, *quality heuristics* specify which of the chosen blade designs are of sufficient quality to be reported.

We remark that the realization of these heuristics will vary depending on the wishes of the vibratory bowl designer. In our implementation, our first priority is the accurate representation of distinct feeding behaviors expressed in S ; our second priority is to report the blade designs that are least sensitive to tolerances in part motion and shape.

We apply a straightforward subdivision heuristic: each cell of S is represented by one blade—recall that each cell of S is comprised of blade designs of equal feeding behavior.

Intuitively, given a cell of S , our representation heuristic aims to select the blade design for which the number of neighboring blade designs of equal feeding behavior is maximized in all three dimensions. In general, this concept is realized by selecting the center of an inscribed shape of each cell. The type (e.g. a box or ellipsoid) and aspect ratio of this inscribed shape should reflect the dependencies between and relative importance of φ , h and w with respect to insensitivity to motion tolerances. These dependencies and relative importances are, however, a subject of future (experimental) study. For our test parts, we implement the representation heuristic by taking the center of each cell with respect to h , and subsequently taking the center of the resulting 2D cell cross-section with respect to φ and w .

Lastly, our quality heuristic discards the blade designs that do not satisfy a minimum degree of insensitivity. For our three test cases, we discard the blade designs with less than 0.015 inch, 0.015 inch or 1° distance to the cell boundary in the width, height and angle dimensions, respectively.

As an advanced note on observed discrepancies, we found during experiments that certain quasi-statically stable track poses are practically unstable. In order to generate a design report usable for our physical experiments, such track poses should not be considered by the quasi-static design algorithm: we marked a track pose practically unstable if it was not able to travel the feeder track (without blade module) at least once out of three test runs.

IV. PHYSICAL EXPERIMENTS

In this section, we discuss the hardware implementation of the blade, illustrate the experimental test cases and present resulting observations and conclusions.

A. Reconfigurable Modular Hardware

The model 5300A.1 (T-18 – Automation Devices, Inc) vibratory inline-feeder, about 13 inches in length, provides the driving force at 120 Hz. A metal track floor is mounted on the T-18, on which in turn a track wall is mounted. For the attachment of the wall, we used a “rail system” that consists of three parallel narrow slots onto which the wall is bolted. Similarly, the track wall contains rails onto which the blade is attached (Fig. 5). These two rail systems enable us to adjust w and h , respectively, in a continuous manner. The blade angle φ can be stepwise adjusted by mounting different blade modules onto the track wall; modules with a blade angle of 20° , 27° , 33° , 37° , 49° , 60° and 70° were made. As a practical addition, we note the “knocker” wiper to the side of the track (Fig. 1); this ensures that rejected parts hanging off the side of the track are forced off. Lastly, the track tilt that ensures the stable wall and blade contact of part poses is 5° .

B. Experiments

Employing this hardware setup, we have conducted an initial series of physical experiments to gain a better understanding of the effectiveness of the quasi-static design

TABLE I
EXPERIMENTAL RESULTS PART A

Exp. #	1	2	3	4	5	6	7	8	9
φ (deg.)	20	27	33	37	49	60	27	27	27
h (in.)	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54
w (in.)	0.00	0.00	0.00	0.00	0.00	0.00	-0.28	-0.07	0.10
Des. #	2	2	2	2	1	3	2	2	2
4.1 (%)	13	40	53	53	66	0	0	33	0
4.2 (%)	100	100	93	93	0	0	0	66	100
4.3 (%)	100	100	100	100	100	0	0	100	100
4.4 (%)	100	100	100	100	66	0	0	100	100
Runs	15	15	15	15	3	3	6	3	3
Feeds	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes

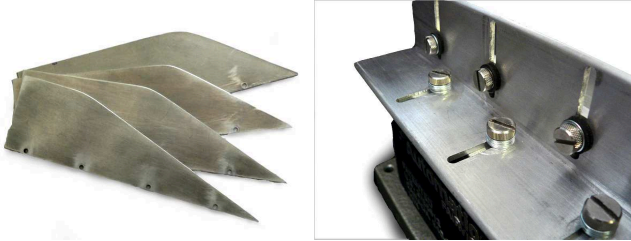


Fig. 5. Photos of four blade modules with different blade angles (left), and a view of the hardware at the “backside” of the track wall, showing the two rail systems for continuous adjustment of the blade width and height.

algorithm. In Tables I and II, we present a selection of the results of these experiments, which are conducted with three test parts: part A (Fig. 4), part C (Fig. 6a) and part B (Fig. 6b). For lack of space, we refer to the on-line design reports [9] to gain insight into the intended operation of the evaluated blade designs. Lastly, important to note is that although often the blade settings of the hardware and the design report do not exactly match, they do feature the same feeding behavior (subsection III-C).

We consider the presentation of the data in Tables I and II. Each column describes an experiment with one physical blade setup. The applied hardware settings are specified in the ‘ φ ’ (in degrees), ‘ h ’ (in inches) and ‘ w ’ (in inches) rows; the ‘Des(ign) #’ row gives the sequence number of corresponding blade design in the on-line design report. For each experiment, the percentage of runs for which a track pose $a.b$ is fed is specified in the rows below; the track poses that are rejected for all runs are not listed. Bold marking of a result (percentage) indicates that the corresponding track pose should be fed according to the design report; i.e., for an ideal match between the design report and the physical experiments, the bold results should be 100% and the rest should 0%. Furthermore, the ‘Feeds’ row specifies whether the part was fed, where bold markings indicate that the output (blade) pose matches the predictions of the design report.

In general, the experiments suggest that although the quasi-static algorithm provides an accurate foundation for blade design—the feeding behavior of the majority of track poses is correctly predicted—the observed discrepancies must be taken into account to increase the reliability of the algorithm. The following section includes a more detailed

TABLE II
EXPERIMENTAL RESULTS PART B (LEFT) AND PART C (RIGHT)

Exp. #	10	11	Exp. #	12	13	14
φ (deg.)	20	20	φ (deg.)	20	20	20
h (in.)	0.42	0.20	h (in.)	0.30	0.32	0.15
w (in.)	0.27	0.25	w (in.)	0.20	0.20	0.20
Des. #	6	4	Des. #	4	4	2
1.4 (%)	33	0	1.2 (%)	100	100	0
1.5 (%)	33	0	1.3 (%)	100	100	0
6.1 (%)	100	100	5.4 (%)	100	0	100
6.2 (%)	100	33				
6.3 (%)	100	33				
6.4 (%)	100	0				
Runs	3	3	Runs	3	3	3
Feeds	No	Yes	Feeds	No	Yes	Yes

report on the observed discrepancies.

V. AUGMENTED DESIGN ALGORITHM

Let us first discuss the concept behind the augmented design algorithm. We recall that the quasi-static algorithm consists of two consecutive phases: firstly, the modeling phase, which constructs the critical surfaces, each of which captures the effect of all blade designs for one specific track pose; and secondly, the extraction phase, which extracts S_{qs} from the resulting arrangement of critical surfaces. Between these phases, we insert a new phase that we refer to as the *augmenting phase*. Intuitively, taking the set of critical surfaces as input, this intermediate phase “pads” each critical surface with a layer of uncertainty. Afterward, these padded critical surfaces are given to the extraction phase that subsequently extracts S_{dy} ; this process is similar to the extraction of S_{qs} . We note that after the augmented algorithm, the selection heuristics can again be applied to generate a design report.

We assume to be able to query external physics software—in order to assess the amount of padding added to the critical surfaces. Development of a concrete approach for this approximation process is a subject of future research.

As noted before, the set S_{dy} is in essence the result of the complete quasi-static algorithm, augmented by incorporating the approximate effect of the observed discrepancies. Clearly, the solution set S_{dy} will in general (partially) differ from S_{qs} . A more subtle difference, however, comes from the potential ability of the augmented algorithm to add probability information to valid blade designs. That is, while a blade design in S_{qs} is “naively” assumed to always feed each track pose of a certain set of track poses, a blade design in S_{dy} feeds a certain expected percentage of the occurrences of this track pose; the latter is a key ingredient for an approximation of the practical feed-rate of a given blade design.

The part-blade interaction can be split into four consecutive phases: P moves along the track in a track pose, the blade reorients P from track pose to blade pose, P moves along the blade in a blade pose, and the blade (possibly) rejects P . We discuss how each of these

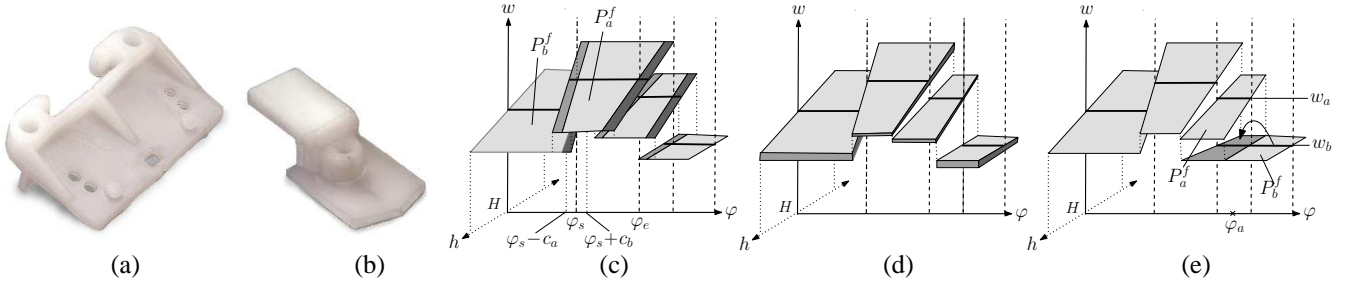


Fig. 6. (a, b) Photos of ‘part C’ and ‘part B’. (c, d, e) Slices of a critical surface highlighted at $h = H$: (c) the dark gray areas illustrate “bands” of uncertainty in reorientation from track to blade pose, (d) expansion of single w_c value to an range of uncertainty in critical width values for each φ, h -pair, (e) the dark gray surface captures the uncertainty in stability of blade pose P_a^f .

phases of part-blade interaction is affected by the observed discrepancies. Although new experiments may reveal yet other discrepancies, we believe the following to provide a fairly complete overview.

Track Poses

As mentioned in section III-D, we observe that quasi-statically stable track poses can be practically unstable. Such an event in general occurs when the projection of the center of mass is close to the boundary of the (convex hull) face resting on the track floor; e.g. consider track pose 6.1 of part A [9]. Only the practically stable track poses are to be considered by the augmented algorithm as a basis for further calculation. As a possible alternative to the application of physics software, existing literature proposes several approaches [5], [12] to identify practically stable track poses.

Pseudo-stable stable track poses are a special case of unstable track poses. We observed this part behavior in experiment 10: pseudo-stable track poses 1.4 and 1.5 of part B periodically shift their orientation toward that of (practically unstable) track pose 4.3. This behavior can result in unpredicted feeding of track poses such as observed for 1.4 and 1.5. The phenomenon of pseudo-stable track poses is a subject of future research.

Reorientation of Track Pose to Blade Pose

An observed discrepancy is uncertainty in the reorientation of track pose to blade pose by a given blade design. We can observe this in experiment 1 through 4: the quasi-static algorithm specifies that design #2 reorients track pose 4.1 in clockwise direction (when seen from above), which subsequently results in the rejection of 4.1; while for a certain percentage of experimental runs, track pose 4.1 is observed to be fed by design #2 because 4.1 is reoriented counter-clockwise instead of clockwise.

We outline the extension of the quasi-static algorithm, which adds an approximation of the effect of uncertainty on reorientation. Considering the critical surface of a track pose P_e^f , let $[\varphi_s, \varphi_e]$ be the blade angle interval at $h = H$, for which $B(\varphi, H)$ with $\varphi \in [\varphi_s, \varphi_e]$ reorients P_e^f to blade pose P_a^f . In the augmented algorithm, the precisely defined boundary value φ_s is replaced by a range of blade angles $[\varphi_s - c_a, \varphi_s + c_b]$; blade designs $B(\varphi, H)$ with $\varphi \in [\varphi_s - c_a, \varphi_s + c_b]$ can reorient P_e^f to either blade

pose P_a^f or P_b^f (Fig. 6c), where $a = \theta_i$ and $b = \theta_{i-1}$ (sub-section II-D). A similar idea can be applied to both φ_s and φ_e for any h , resulting in “bands” in the critical surface for P_e^f can be reoriented both clockwise and counter-clockwise. We remark that the width of these bands, i.e. the values of c_a and c_b , are probable to vary for each boundary and h -value.

Blade Poses

We present three observed discrepancies with respect to the motion of the part along the blade.

1. *Uncertainty in stability of blade contact:* The blade contact (which discretizes the θ orientation) of a blade pose P_θ^f may be practically unstable; in such event, P_θ^f topples over and stabilizes onto a neighboring edge, hence changing its θ orientation. We observe this in experiment 10. The quasi-static algorithm specifies that blade design #6 reorients track poses 6.1 and 6.2, 6.3, 6.3 to blade poses P_a^f and P_b^f , respectively, of which only the latter is output. Experimentally, however, 6.1 is fed by design #6. This is explained by the observation that P_a^f , which would be rejected, is practically unstable and subsequently rotates to P_b^f , which is output.

Let us consider track pose P_e^f and blade height $h = H$. Also, let P_a^f and P_b^f be blade poses with $a = \theta_i$ and $b = \theta_{i+1}$; the stability of the blade contact of P_a^f is uncertain such that P_a^f may rotate to P_b^f . Consequently, although the quasi-static algorithm specifies that a given blade $B(\varphi_a, h)$ reorients P_e^f to P_a^f , it may occur that $B(\varphi_a, h)$ effectively reorients P_e^f to P_b^f . Next, we recall that for any given φ and h , the critical width is determined by the blade pose to which P_e^f is reoriented. Hence follows that for φ_a and H , there now exists two w_c -values: one of P_a^f and one of P_b^f —these are shown as w_a and w_b in Fig. 6e.

Intuitively, by propagating the uncertainty in stability of blade contacts, the critical surface becomes double-layered in certain regions. A point above such a double-layered region corresponds to a blade that output two different blade poses.

2. *Unstable floor contact:* Although the floor contact of a track pose may be practically stable, we observe that a floor contact can become practically unstable once the part starts moving along the blade. An example of such event is observed in experiment 13: the quasi-static algorithm specifies that blade design #4 reorients track pose 5.4 of part C to a blade pose, P_θ^f , which is subsequently fed; however,

in the experiments we observe that the floor contact of P_θ^f is practically unstable against blade design #4, resulting in the rejection of 5.4. The extension of the quasi-static algorithm to take this behavior into account is subject to future research.

3. *Hampered part motion*: High friction between the part and the blade or track floor can hamper the motion of P_e^f and result in undesirable feeding behavior. A noticeable source of high friction observed in the experiments are faces of the part that are near parallel to the track floor. Intuitively, interaction between a blade and such a face causes the part to get “wedged” between the track floor and the blade. We observed this behavior in experiment 12, 13 and 14 with e.g. track poses 1.2 and 1.3.

The extension of design algorithm to minimize such events can be outlined as follows. For each track pose, we propose to combine the friction coefficient of the material and the slope of each face into a single coefficient. The motion of a pose along the blade is likely to be hampered, when the pose rests against the blade with a face featuring a coefficient above some threshold. Intuitively, for each track pose the set of such faces maps to a set of blade designs, the union of which should be subtracted from S_{dy} to make hampered part motion less likely.

Rejection of Blade Poses

A last observed discrepancy is uncertainty in the rejection of a blade pose by a given blade design. We observe this in experiment 7: the quasi-static algorithm specifies that blade design #2 feeds track poses 4.2, 4.3 and 4.4 of part A; however, the blade setting in experiment 7, which features the same feeding behavior as blade design #2, is observed to reject track poses 4.2, 4.3 and 4.4. Intuitively, for a given P_e^f , this uncertainty replaces the one critical blade width w_c value by a range of critical w -values for each φ and h (Fig. 6d). We note that this uncertainty is probable to vary for different blade poses.

VI. CONCLUSION

We have presented a series of new developments on the design of blades. The concept and quasi-static design algorithm of the blade was introduced in [10]. In this paper, we have developed a software implementation of this quasi-static design algorithm, and presented a reconfigurable algorithm hardware implementation. Employing both hard- and software, we have conducted physical experiments and have reported our findings. Our findings suggest that, although it provides an accurate foundation for blade design, the reliability of the design algorithm can be improved. To this end, we consider the part behaviors that are observed to cause discrepancies between the results of the quasi-static algorithm and the physical experiments. An augmented design algorithm is presented, which extends the quasi-static algorithm by taking the effect of these discrepancies on the part-blade interaction into account.

Our aim in this paper has been to take an important step toward the practical application of complete design algorithms in vibratory bowl design. Near future research

to further consolidate this step includes additional physical experiments to complete our understanding of the physical part-blade interaction. Further research is also needed on the subjects of selection heuristics and the assumption that the part does not change its the floor contact while moving along the track or interacting with the blade.

The intertwining of the complete quasi-static design algorithm and physics software promises to be an interesting direction of further study. We conceive that this approach combines the advantages of two worlds: the physics software allows for assessing aspects of relevant part-blade interaction that cannot be expressed in a quasi-static model; while the exact quasi-static algorithm ensures feasible computation times by providing a complete basis solution into which the aforementioned assessed aspects can be “plugged”.

A final challenging direction of further research is the classification of parts that can be fed by a blade. We conjecture that symmetry and regularity in part shapes play a key role in this subject.

REFERENCES

- [1] P.K. Agarwal, A.D. Collins, and J.L. Harer. Minimum trap design. *IEEE ICRA*, pages 2243–2248, 2001.
- [2] D. Berkowitz and J. Canny. Designing part feeders using dynamic simulation. *IEEE ICRA*, pages 1127–1132, 1996.
- [3] R.-P. Berretty, K. Goldberg, M.H. Overmars, and A.F. van der Stappen. Trap design for vibratory bowl feeders. *International Journal of Robotics Research*, 20:891–908, 2001.
- [4] R.-P. Berretty, K.Y. Goldberg, M.H. Overmars, and A.F. van der Stappen. Algorithms for fence design. *Robotics, the algorithmic perspective*, pages 279–295, 1998.
- [5] G. Boothroyd. *Automatic Assembly and Product Design*. Taylor & Francis Ltd, 2005.
- [6] M. Caine. The design of shape interactions using motion constraints. *IEEE ICRA*, pages 366–371, 1994.
- [7] A. Christiansen, A. Edwards, and C. Coello. Automated design of parts feeders using a genetic algorithm. *IEEE ICRA*, pages 846–851, 1996.
- [8] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry – Algorithms and Applications*. 1997.
- [9] O.C. Goemans, Marshall T. Anderson, K. Goldberg, and A.F. van der Stappen. Design reports, 2007. <http://www.cs.uu.nl/people/onno/reports/index.html>.
- [10] O.C. Goemans, K. Goldberg, and A.F. van der Stappen. Blades: a new class of geometric structures for feeding 3d parts on vibratory tracks. *IEEE ICRA*, pages 1730–1736, 2006.
- [11] O.C. Goemans, A. Levandowski, K. Goldberg, and A.F. van der Stappen. On the design of guillotine traps for vibratory bowl feeders. *IEEE CASE*, pages 79–86, 2005.
- [12] K. Goldberg, B. Mirtich, Y. Zhuang, J. Craig, B. Carlisle, and J. Canny. Part pose statistics: Estimators and experiments. *IEEE Trans. on Robotics and Automation*, 15(5):849–857, 1999.
- [13] M. Jakiela and J. Krishnasamy. Computer simulation of vibratory part feeding and assembly. *2nd Int. Conf. on Discrete Element Methods*, pages 403–411, 1993.
- [14] L. Lim, B. Ngoi, S. Lee, S. Lye, and P. Tan. A computer-aided framework for the selection and sequencing of orientating devices for the vibratory bowl feeder. *Int. J. of Production Research*, 32(11):2513–2524, 1994.
- [15] G. Maul and M. Thomas. A systems model and simulation of the vibratory bowl feeder. *2nd International Conference on Discrete Element Methods*, 16(5):309–314, 1997.
- [16] A. Rao and K. Goldberg. Friction and part curvature in parallel-jaw grasping. *Journal of Robotic Systems*, 12(6):365–382, 1995.
- [17] J.M. Selig and J.S. Dai. Dynamics of vibratory bowl feeders. *IEEE ICRA*, pages 3299–3304, 2005.
- [18] R. Silversides, J.S. Dai, and L.D. Seneviratne. Force analysis of a vibratory bowl feeder for automatic assembly. *ASME: Journal of Mechanical Design*, 127(4):637–645, 2005.