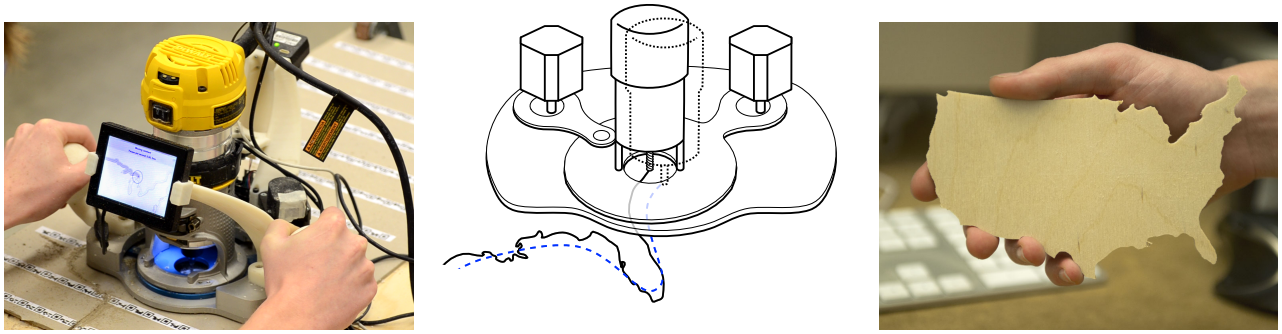
Position-Correcting Tools for 2D Digital Fabrication

Alec Rivers
MIT CSAIL

Ilan E. Moyer
MIT

Frédo Durand
MIT CSAIL



(a) (b) (c)

Figure 1: Overview: (a): *A position-correcting tool. The device consists of a frame and a tool (in this case a router) mounted within that frame. The frame is positioned manually by the user. A camera on the frame (top right in the figure) is used to determine the frame's location. The device can adjust the position of the tool within the frame to correct for error in the user's coarse positioning.* **(b):** *To follow a complex path, the user need only move the frame in a rough approximation of the path. In this example, the dotted blue line shows the path that the tool would take if its position were not adjusted; the black line is its actual path.* **(c):** *An example of a shape cut out of wood using this tool.*

Abstract

Many kinds of digital fabrication are accomplished by precisely moving a tool along a digitally-specified path. This precise motion is typically accomplished fully automatically using a computer-controlled multi-axis stage. With that approach, one can only create objects smaller than the positioning stage, and large stages can be quite expensive. We propose a new approach to precise positioning of a tool that combines manual and automatic positioning: in our approach, the user coarsely positions a frame containing the tool in an approximation of the desired path, while the device tracks the frame's location and adjusts the position of the tool within the frame to correct the user's positioning error in real time. Because the automatic positioning need only cover the range of the human's positioning error, this frame can be small and inexpensive, and because the human has unlimited range, such a frame can be used to precisely position tools over an unlimited range.

Keywords: personal digital fabrication, personalized design, CAD, CAM

Links: DL PDF

1 Introduction

Personal digital fabrication endeavors to bridge the gap between computer graphics and the real world, turning virtual models into physical objects. Novel software modeling allows users to create unique objects of their own design, e.g. [Mori and Igarashi 2007; Kilian et al. 2008; Lau et al. 2011; Saul et al. 2011], which can then be fabricated using 2D devices such as laser or water jet cutters, or 3D devices such as 3D printers and computer numerical control (CNC) mills. While rapid prototyping machines are dropping in price, affordable tools have severe size limitations because of the expense of a precise and long-range positioning system. As an illustration, a 2' $\times$ 1.5' ShopBot CNC mill costs approximately $6,000, while a 5' $\times$ 8' ShopBot mill costs over $20,000 [ShopBot Tools].

We aim to reduce the cost of digital fabrication for the domain of 2D shapes while simultaneously removing constraints on range. Our central idea is to use a hybrid approach to positioning where a human provides range while a tool with a cheap short-range position-adjustment enables precision. Given an input 2D digital plan such as the outline of a shape, the user manually moves a frame containing a tool in a rough approximation of the desired plan. The frame tracks its location and can adjust the position of the tool within the frame over a small range to correct the human's coarse positioning, keeping the tool exactly on the plan (Figure 1). A variety of tools can be positioned in this manner, including but not limited to a router (which spins a sharp bit to cut through wood, plastic, or sheet metal in an omnidirectional manner) to cut shapes, a vinyl cutter to make signs, and a pen to plot designs.

In this approach, the core challenges are localization (determining the current position of the tool) and actuation (correcting the tool's position). For localization, we use computer vision and special markers placed on the material. For actuation, we present a two-axis linkage that can adjust the position of the tool within the frame. We also describe an interface for guiding the user using a screen on the frame, which illustrates the tool's current position relative

to the plan. We show an example of a device (Figure 1), measuring roughly 13" x 10" x 9", that uses our approach and can be fitted with a router or a vinyl cutter, and show results that can be achieved with these tools when they are positioned with our computer-augmented approach.

2 Related Work

Recent work on personal digital fabrication has yielded interfaces that integrate fabrication considerations with design, allowing fabrication-conscious design of a variety of material and object types such as plush toys [Mori and Igarashi 2007], chairs [Saul et al. 2011], furniture [Lau et al. 2011], shapes made out of a single folded piece of material [Kilian et al. 2008], and paneled buildings [Eigensatz et al. 2010]. Other papers explore how to generate designs with desired physical properties, such as deformation characteristics [Bickel et al. 2010], appearance under directed illumination [Alexa and Matusik 2010], and subsurface scattering [Dong et al. 2010; Hašan et al. 2010].

When it comes to fabricating objects from these designs, the most widely used devices are 3D printers, laser cutters, and CNC milling machines. Recently, a variety of efforts growing out of the DIY community have sought to reduce the cost of 3D printers [MakerBot Industries ; Drumm 2011; Sells et al. 2009] and CNC mills [Hokanson and Reilly ; Kelly]. These projects typically provide relatively cheap kits for entry-level devices. However, as with professional models, positioning is done with a multi-axis stage, and the trade-off between cost and range remains.

Our computer-augmented positioning approach removes the limitation on range of traditional gantry-based positioning technologies. To do so, it relies on accurately detecting the position of the frame in real time. A variety of approaches to real-time localization have been employed over the years, from global-scale GPS [Getting 1993] to local-scale systems based on radio and ultrasonic signals [Priyantha et al. 2000]; an overview is given in a survey by Welch and Foxlin [2002].

Our approach to localization is based on computer vision. Computer vision has been widely used for position tracking in the context of motion capture (see Moeslund et al. [2006] for a survey). These setups typically use stationary cameras tracking a moving object, though recently Shiratori et al. [2011] proposed a system in which cameras are placed on the human and track the environment. In our approach, the camera is on the tool and tracks the material over which it moves, first stitching frames together to make a map of the material (see Zitova and Flusser [2003] and Szeliski [2006] for surveys of image registration and stitching techniques) and then using that map to perform localization. This approach has been used before, with some differences, in a recent new peripheral, LG's LSM-100 scanner mouse [LG ; Zahnert et al. 2010], which is a mouse that can scan a document it is passed over. Our implementation differs from theirs in that we use only a camera (no optical mice), capture a wider area of the material in each frame, and use high-contrast markers placed on the material to allow capture of untextured materials.

Computer vision has previously been applied to CNC manufacturing, for example to monitor for mistakes [Al-Kindi et al. 1993], or to precisely align a tool path to a piece of material [Techno CNC Router Systems]. These approaches, however, do not re-imagine the fundamental approach or form factor of a table-based, fully automatically-positioned CNC device. A hybrid approach has been taken in the case of computer-assisted surgical devices [Kragic et al. 2005; Mako Surgical], for example by using a robot to recreate a manual tool motion at a smaller scale for microsurgery. However, the motivation in these cases is to "take advantage of robotic speed and precision, but avoid the difficulties of full autonomy by retaining the human 'in the loop' for essential decision making and/or physical guidance" [Kragic et al. 2005]. By comparison, our goal is to leverage the human's mechanical range, rather than decision making power or guidance, to enable a new form factor and approach to a task that is currently fully automated.

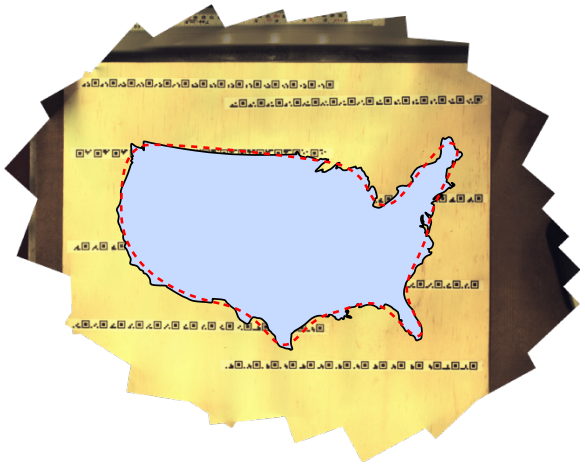


Figure 2: Map: *A scanned map with a plan registered to it. The red dotted line indicates a path that a user could conceivably follow to cut out the shape.*

3 Localization

To keep the tool on the plan as closely as possible, the tool must detect its current position accurately, robustly, and with low latency.

We considered a variety of localization systems, eventually settling on a simple computer vision-based approach, in which a camera on the frame of the device tracks high-contrast markers placed in an arbitrary pattern on the material. A map of the material (Figure 2) is first built by passing the device back and forth over the material to be cut; then, images from the camera are compared to this map to determine the device's location. This approach was chosen for a variety of reasons: it can achieve very high accuracy; it always remains calibrated to the material, as the markers are on the material itself (as opposed to external beacons, which can become uncalibrated); it does not require excessive setup; the hardware required is relatively inexpensive; and it can be implemented using standard computer vision techniques. Building the map is fast and easy.

We considered using the camera to track just motion, as in an optical mouse, but this approach would be subject to drift. An alternative would be to draw the plan on the material itself, e.g. with a pencil, and then track that, but that would require additional work on the part of the user and could introduce error.

3.1 High-contrast markers

We leverage specially-printed tape marked with high-contrast patterns to make it possible to track materials that have no visual features of their own (such as sheet metal or plastic) and to increase robustness under varying lighting conditions. This tape is applied before map-making, in any pattern so long as some tape is visible from every position that the device will move to, and can be removed when the job is complete. The tape consists of many Quick Response (QR) code-like markers [Denso-Wave Incorporated] in a row, each consisting of an easily-detectable box-within-box pat-

tern we call an "anchor" and a 2D barcode that associates a unique number with the anchor (see Figure 3). As long as four of these markers are visible at any time (which is typically the case even if only a single piece of tape is visible), the device is able to locate itself. The redundancy of the markers means that it does not matter if some are occluded (e.g. by sawdust) or obliterated by the tool itself. Note that these markers function just as features – their positions are not assumed before mapping, and they need not be laid out in any specific pattern.

Figure 3: Markers: *A sequence of markers, with values 1000 to 1006, such as would be printed on a strip of tape. In our current implementation, markers are printed at a size of roughly 0.8"×0.4". This is small relative to the area of the material the camera can see at once (roughly 8"×6").*

3.2 Image processing

The core operations used during locating and building a map are detecting markers in an image and registering one set of markers onto another.

Detecting markers To detect markers, the frame is first binarized using the Otsu method [1979], and then rectified to a top-down orthographic view based on a one-time calibration of the camera relative to the flat plane on which the tool sits. Anchors are extracted using a standard approach to QR code reading: first, horizontal scanlines are searched for runs of alternating pixel colors matching the ratio of 1:1:3:1:1, as will always be found at an anchor. Locations that match this pattern are checked for the same pattern vertically. Locations that match horizontally and vertically are floodfilled to confirm the box-within-box pattern. Once anchors have been extracted, each anchor is experimentally matched with the nearest anchor, and the area in between is parsed as a barcode. Barcode orientation is disambiguated by having the first bit of the 2D barcode always be 1 and the last bit always be 0. If the parsed barcode does not match this pattern, the next-nearest anchor is tried. If neither matches, the anchor is discarded. If the pattern is matched, the barcode's value is associated with the first anchor and that anchor's position is added to the list of detected markers.

Matching sets of markers To match two sets of markers, we find all pairs of two markers, one from each set, that share the same ID. If there are at least four such pairs, we run RANSAC [Fischler and Bowles, 1981] to find the Euclidean transformation that maps the positions of the pairs' markers in the first set to the corresponding positions in the second set, with a tolerance of 5 millimeters. While only two pairs are sufficient to determine a Euclidean transformation, we set the minimum number of inliers to four to prevent false positives. The resulting least-squares transformation matrix establishes a relative position constraint between the sets.

3.3 Building a map

Mapping is done by stitching together video frames into a 2D mosaic (Figure 2) as the user passes the device back and forth over the material. To reduce memory loads, we retain only frames that overlap with the previously retained frame by less than 75%, as computed after determining the frame's position relative to all other frames as as described below.

We use a simple method to stitch images together. Each frame's current position and orientation in the map is expressed as a 3×3 transformation matrix. The first frame is assigned the identity matrix. When a new frame is received, a match is attempted with all previous frames. Every successful match generates one relative position constraint, also expressed as a 3×3 matrix. The new frame's transformation matrix is obtained by multiplying each matched frame's transformation matrix with the corresponding constraint matrix, and averaging the resulting matrices. The new frame's transformation matrix is then orthonormalized.

Because frames' positions are constrained only relative to overlapping frames, the transformation between two non-overlapping frames depends on the relative constraints of some number of in-between frames. This transformation is less accurate as the number of intermediate connections increases. This may cause discontinuities where the sequence of frames loops back on itself. We smooth out these discontinuities by iteratively reapplying relative position constraints for all frames once all images have been acquired, until the system converges to a stable configuration, similar to bundle adjustment [Triggs et al. 2000]. In each iteration of smoothing, each frame computes a new transformation matrix by multiplying each connected frame's transformation matrix from the previous iteration with the corresponding constraint matrix, and averaging the resulting matrices across all constraints. As before, the new transformation matrix is then orthonormalized. This process always converges in practice.

Once the map is complete, a super-list of markers for the entire map is generated from the markers in input images by averaging the map-space positions of markers that share the same ID. This global list of known positions for each marker ID is used to localize new images when the device is in use.

When preparing to cut a shape, the user registers a shape onto this 2D map. Having the map of the material makes it trivial to visually align the plan with features of the material. With a traditional CNC machine, this would typically require careful calibration.

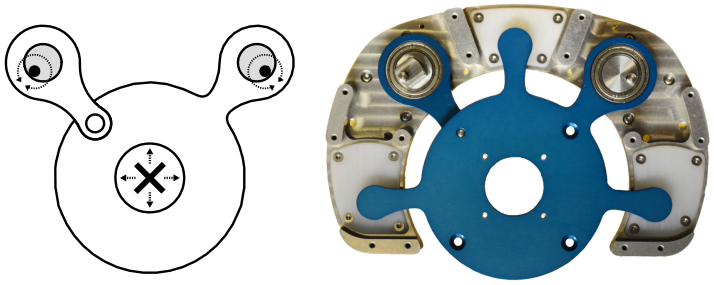
3.4 Localization using the map

Once the map has been created, registering a new image to the map is straightforward. Markers are detected as above and matched to the global list of markers using the same RANSAC algorithm as above. An image from the camera can be registered to a map in $\sim$4 milliseconds total on a standard laptop. Although localization exhibits strong time coherence, thanks to the low cost of processing we can afford to solve the system from scratch at every frame.

4 Actuation

Once the location of the frame is known, the tool must be repositioned within the frame to keep it on the plan. This task can be broken down into the control challenge of determining where within the frame to move (as there are usually many possible positions that lie on the plan) and the mechanical engineering challenge of building an accurate, responsive, and low-cost position-adjusting actuation system.

The range of our positioning linkage was determined by first attempting to move the frame along a 2D plan as closely as possible by hand. We found that when provided with accurate location information relative to the plan a user can keep the tool within 1/8" of the plan, even when cutting wood. Having accurate location information allows for greater precision than normal freehand positioning. To allow for a safety margin and increase ease of use, we doubled this value to arrive at the goal of being able to correct errors up to 1/4" (i.e. having a range circle with a 1/2" diameter).



(a) (b)

Figure 4: Positioning linkage: (a): *Our linkage converts the rotary motion of two shafts (filled circles) into translation of a tool (cross) mounted on a sliding stage. This is achieved using eccentrics (shaded circles), which are circular disks rotating about the off-center shafts to produce linear displacement in fitted collars. To properly constrain the degrees of freedom of the stage, one eccentric is directly connected to the stage while the other is connected via an additional hinge.* **(b):** *A photo of the actual linkage.*

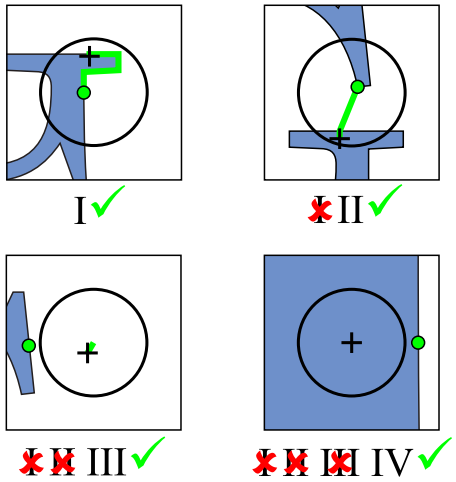


Figure 5: Freeform motion paths: *Each box illustrates a case in which a different path (described below) is used, due to the higher-preference paths being infeasible. In each box, the cross is the current position of the tool, the circle is the range of the positioning system, the green dot is the target position, and the green path is the selected path.*

4.1 Actuation system

The actuation system need only support a small range of motion, as it need only correct the coarse positioning done by the human. This affords the possibility of using a very different design for the positioning system than the multi-axis stage employed by traditional rapid prototyping machines.

Our major mechanical departure from traditional rapid prototyping machines is that we use eccentrics, rather than linear stages, to convert the rotational motion of the motors into linear motion. Eccentrics are circular disks rotating around an off-center shaft. As they are rotated, they produce linear motion in a collar wrapped around the disk. Eccentrics are able to maintain the same low-backlash accuracy of a precision linear stage while being much cheaper. For this, they sacrifice range. However, a linear displacement range of 1/2" is well within the capabilities of an eccentric.

Our design (Figure 4) consists of two eccentrics mounted to the frame and connected to a stage that can slide on its base. The eccentrics are rotated by stepper motors, and by rotating them the stage can be moved within the frame. To properly constrain the stage, one eccentric is connected directly to the stage by a ball bearing coupling, while the other is connected both by a coupling and a hinge.

This linkage design results in a nonlinear relationship between eccentric orientation and stage position: as the tool approaches the edge of its range, increasingly large rotations are necessary to move the stage a fixed amount. We limit stage displacement to ~95% of the maximum range to cut out the positions with extreme nonlinearity. This linkage design also permits backdriving, in that forces acting on the tool can cause the cams to rotate away from their target positions; however, we found that the stepper motors we use (62 oz-in holding torque) are sufficiently powerful to preclude backdriving, even in the presence of significant material forces.

4.2 Following a plan

As the user moves the frame, the device must ensure that the tool stays on the plan. To do this, the path that is to be followed must be first computed (which may not be the same as the plan); then, every frame, given the frame's position, the tool's position within the frame, and the plan, the device must determine where to move the tool within the frame.

For the applications we focus on – routing and vinyl cutting – the user generally wishes to cut a shape out of a piece of material. This means that there will be some areas of the material that are outside the target shape, and which may be cut freely (which we call "exterior material"), while other areas lie inside the target shape and must not be cut ("interior material"). To allow for this distinction, we define a plan as consisting of polygons, which have defined interiors, rather than simply as paths.

In applications such as vinyl cutting, the tool should follow the border of the interior material as closely as possible. When routing, however, the size of the cutting bit must be taken into account, and the tool should move along a path offset from the interior material by the radius of the bit, to leave the actual cut shape as close as possible to the specified plan. We provide an option to set the diameter of the cutting bit and offset the plan polygons accordingly.

We propose two different strategies for moving the tool to keep it on the plan, and will show how each of these is appropriate for a different class of applications.

4.2.1 Constant-speed motion

In the simpler strategy, the tool is moved through the material at as close to a constant rate as possible. This strategy is useful for applications such as routing, in which the material may offer resistance if the tool is moved too quickly and may burn if the tool is moved too slowly.

In this approach, the user decides only what polygon to follow and when to start motion. Thereafter, the software drives the tool around that polygon at a constant rate; the rate is a setting the user can tune (0.2" per second by default). While the tool is moving, the user moves the frame to keep the tool near the center of its range, ensuring that the tool can continue its constant-speed motion without reaching the end of its range. If the tool does reach the end of its range, it must stop until the user catches up.

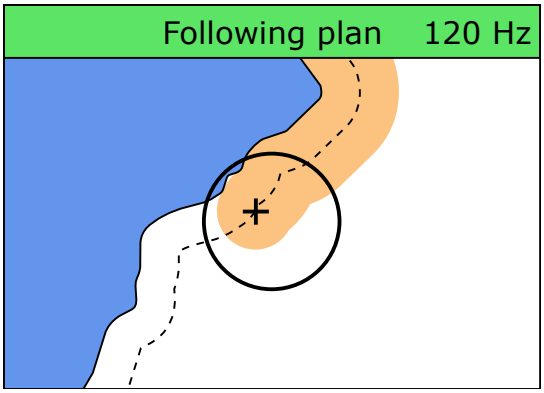


Figure 6: User interface: *This display shows the shapes of the plan (blue polygons); the path that the tool is actually following, which is those shapes offset by the tool's radius (dotted line); the tool's current position (cross); the area cut by the tool (shaded area); and the range of the tool's position correction (black circle). As long as the user keeps the tool path within the correction range, the tool should be able to follow the plan.*

4.2.2 Freeform motion

In the second strategy, the user moves the frame around the plan freely, and the device tries to keep the tool at the point on the plan that most "makes sense" given the user's positioning of the frame. This approach is suitable to applications such as plotting or vinyl cutting in which there is negligible material resistance and no need to move at a constant rate. It can be used to cut shapes much faster than the constant-speed strategy, as the user can slow down only when necessary, such as at sharp turns. In this mode, the amount of error in the tool's positioning is related not to the overall speed of the device, but rather to the speed at which the device is moving away from the plan; roughly speaking, the user should not diverge from the plan faster than 0.2" per second.

The point that the tool is moved to is, generally speaking, the closest point on the border of a plan polygon to the center of the tool's range. However, several considerations make the determination of the path to get to this point complicated. First, the tool should never move through interior material, even if the shortest path from its current position to the target position goes through it. Second, the tool should seek to follow the border of the interior material even when a shorter direct route is possible through exterior material, to avoid skipping over features of the plan.

We aim to account for these considerations while also maximizing the predictability of the tool's motion. We propose a simple strategy in which four possible paths are computed each frame, ranking from most desirable to least desirable, and the most desirable path that is feasible is followed. All seek to move the tool to the target position, which is the closest point on the border of a plan polygon to the center of the tool's range, or to the center of the tool's range itself if the target position is not reachable. These paths, illustrated in Figure 5, are:

I. The path that goes from the tool's position to the nearest point on the border of a polygon, and then walks along the border of that polygon to the target position in whichever direction is shorter. This path is infeasible if it leaves the tool's range or if the target position is on the border of a polygon other than the polygon closest to the tool's position.

II. The path that goes from the tool's position to the nearest exterior material (if it is in the interior material) and then in a straight line to the target position. This path is infeasible if the nearest exterior material is outside the range or the straight line segment passes through interior material.

III. The path that goes from the tool's position to the nearest exterior material (if it is in the interior material) and then in a straight line to the center of the tool's range, stopping whenever interior material is encountered. This path is infeasible if the nearest exterior material lies outside the range of the tool.

IV. Don't move. This path is always feasible.

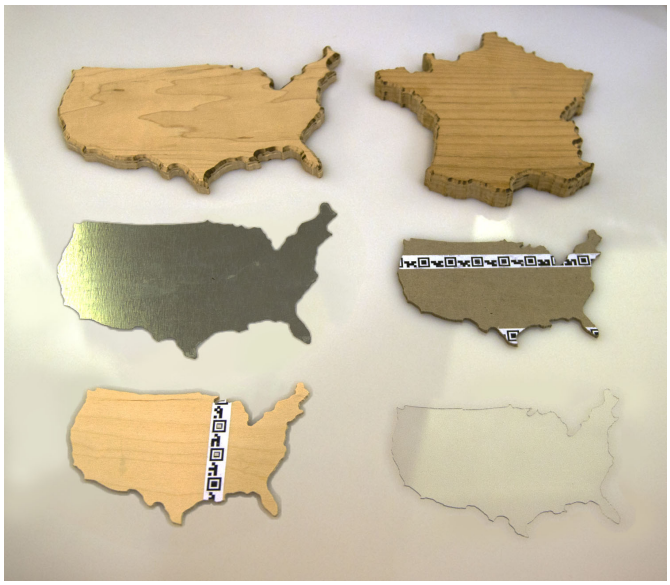


Figure 7: Results: *Several shapes cut out from various materials (clockwise from top left: hardwood, hardwood, paperboard, polycarbonate plastic, plywood, and sheet metal).*

5 Using the tool

The use of the device proceeds as follows: the user places marker tape on the material; the user scans the material; the user registers a plan onto the scanned map of the material; the user uses the device to follow the plan. The user roughly follows the shape of the plan, and the positioning linkage moves the tool to keep it exactly on the plan. In principle, the tool can follow any 2D path. In the application of routing, this means that it can cut out any 2D shape in a single pass, or more complex 2.5D (heightmap) shapes using multiple passes at different depths. Multiple passes can be taken with or without adjusting the plan between passes. For example, we have used multiple passes with the same plan to cut through material thicker than can be cut in a single pass; we have also used different plans to engrave text to a shallow depth on a piece that is then cut out.

5.1 User interface

The user is helped to follow the plan by a display shown on a screen on the tool. This display shows the position of the tool relative to the plan (see Figure 6). In theory, the user's task is to keep the center of the router's motion range as close to the plan as possible. In practice, the user may deviate by as much as the radius of the router's adjustment range.

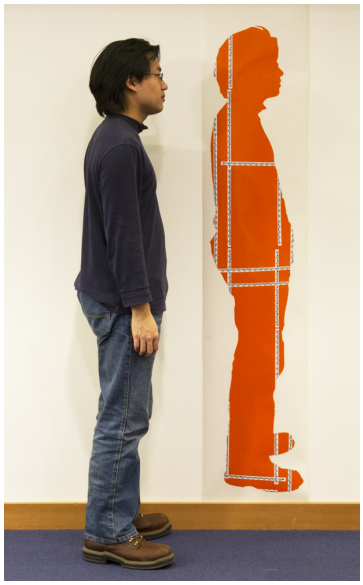


Figure 8: Range: *A full-size vinyl cutout of a human silhouette (5'6" tall), with original.*

We found that this simple interface was able to effectively guide the user in every case we tested. It was easy to follow complex shapes, and thanks to showing the area already cut by the tool, it was possible to interrupt cutting and then return to the correct place.

6 Results

We built a device (Figure 1) that implements the position-correcting system described above. The aluminum base was cut on a CNC mill, while the plastic parts were made with a 3D printer. The device that we built can be mounted with a router or vinyl cutter, and can follow any 2D plan. Figures 1 and 7 show shapes cut out of wood, plastic, paperboard, and sheet metal. Figure 8 demonstrates the range of the device with a full-size vinyl cutout of a human silhouette. Figure 9 shows an example of a cut shape with high-resolution details.

We empirically tested the fidelity of shape reproduction by plotting a complex pattern with a pen mounted as the tool, scanning the result, and measuring deviation from the digital plan (Figure 10). The shape was plotted 6" wide. We fitted a curve to the scanned plot, aligned the plan to that curve, and measured deviation from evenly-sampled points along the drawn shape curve to the nearest point on the plan. The average error was 0.009", with a maximum error of 0.023". The error was small enough that the aligned design always fell within the width of the pen stroke.

7 Conclusion and future work

We have demonstrated a computer-augmented positioning system that avoids the cost-versus-range tension that currently affects rapid prototyping devices, and shown a tool using this approach that combines the unlimited range of a human operator with the accuracy of a computerized positioning system. This device incorporates a computer vision-based system for localization and a specially designed low-cost linkage that can be used to adjust the position of a tool within the device's frame. We have shown how this device can be used with a router and a vinyl cutter to accurately fabricate objects from digital plans.

Figure 9: Fine details: *With a vinyl cutter, the resolution of features is not limited by the width of the bit. Here, we show a 6"-wide sticker with fine details.*

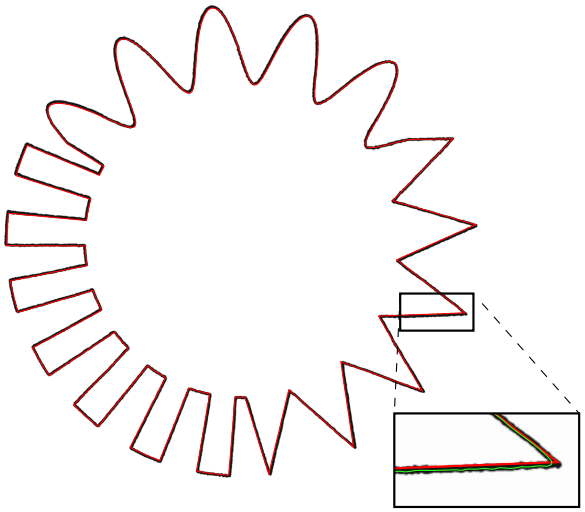


Figure 10: Accuracy: *A scan of a plotted pattern (6" wide) shown with the design that was used to create it (red). The inset shows an expansion of the area of worst error, with the addition of the line fit to the scan for analysis (green). Note that even here the error is only on the order of the width of the pen.*

In future work, we would like to explore applying this type of computer-augmented positioning to a variety of other tools and device form factors. It may also be useful to add automatic Z-axis control to the tool for certain applications, such as when the plan consists of many disconnected subparts. We also wish to explore the possibility of capturing the map using a separate camera looking over the work area, as this would speed up mapmaking and may be useful in cases where many sheets of material are cut in the same physical workspace.

8 Acknowledgments

The authors wish to thank the many people who contributed their time and advice, including Michael Martin, Vladimir Bychkovsky, Adam Saltzman, Aleksandr Kivenson, Peter Brooks, Daniel Reetz, Ron Wikin, Robert Wang, Lewis Girod, Wojciech Matusik, Andrew Adams, Justin Lan, and Skylar Tibbits.

References

AL-KINDI, G., BAUL, R., AND GILL, K. 1993. Vision-controlled CNC machines. *Computing and Control Engineering Journal 4*, 2, 92–96.

ALEXA, M., AND MATUSIK, W. 2010. Reliefs as images. *ACM Trans. Graph. 29*, 4 (July), 60:1–60:7.

BICKEL, B., BÄCHER, M., OTADUY, M. A., LEE, H. R., PFISTER, H., GROSS, M., AND MATUSIK, W. 2010. Design and fabrication of materials with desired deformation behavior. *ACM Trans. Graph. 29*, 4 (July), 63:1–63:10.

DENSO-WAVE INCORPORATED. QR Code Specification. `http://www.denso-wave.com/qrcode/index-e.html`.

DONG, Y., WANG, J., PELLACINI, F., TONG, X., AND GUO, B. 2010. Fabricating spatially-varying subsurface scattering. *ACM Trans. Graph. 29*, 4 (July), 62:1–62:10.

DRUMM, B., 2011. Printrbot. `http://www.printrbot.com/`.

EIGENSATZ, M., KILIAN, M., SCHIFTNER, A., MITRA, N. J., POTTMANN, H., AND PAULY, M. 2010. Paneling architectural freeform surfaces. *ACM Trans. Graph. 29*, 4 (July), 45:1–45:10.

FERRAIOLO, J., FUJISAWA, J., AND JACKSON, D., 2003. Scalable Vector Graphics (SVG) 1.1 Specification. World Wide Web Consortium, Recommendation REC-SVG11-20030114.

FISCHLER, M. A., AND BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24*, 6 (June), 381–395.

GETTING, I. 1993. Perspective/navigation-The Global Positioning System. *IEEE Spectrum 30*, 12, 36–38, 43–47.

GROSS, M. 2009. Now more than ever: Computational thinking and a science of design. *Japan Society for the Science of Design 16*, 2, 50–54.

HAŠAN, M., FUCHS, M., MATUSIK, W., PFISTER, H., AND RUSINKIEWICZ, S. 2010. Physical reproduction of materials with specified subsurface scattering. *ACM Trans. Graph. 29*, 4 (July), 61:1–61:10.

HOKANSON, T., AND REILLY, C. DIYLILCNC. `http://diylilcnc.org/`.

KELLY, S. Bluumax CNC. `http://www.bluumaxcnc.com/Gantry-Router.html`.

KILIAN, M., FLÖRY, S., CHEN, Z., MITRA, N. J., SHEFFER, A., AND POTTMANN, H. 2008. Curved folding. *ACM Trans. Graph. 27*, 3 (Aug.), 75:1–75:9.

KRAGIC, D., KRAGIC, D., MARAYONG, P., LI, M., OKAMURA, A. M., AND HAGER, G. D. 2005. Human-machine collaborative systems for microsurgical applications. *International Journal of Robotics Research 24*, 731–741.

LANDAY, J. A. 2009. Technical perspective: Design tools for the rest of us. *Commun. ACM 52*, 12 (Dec.), 80–80.

LAU, M., OHGAWARA, A., MITANI, J., AND IGARASHI, T. 2011. Converting 3D furniture models to fabricatable parts and connectors. *ACM Trans. Graph. 30*, 4 (Aug.), 85:1–85:6.

LG. LSM-100. `http://www.lg.com/ae/it-products/external-hard-disk/LG-LSM-100.jsp`.

MAKERBOT INDUSTRIES. MakerBot. `http://www.makerbot.com/`.

MAKO SURGICAL. RIO Robotic Arm Interactive System.

MOESLUND, T. B., HILTON, A., AND KRÜGER, V. 2006. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding 104*, 2-3 (Nov.), 90–126.

MORI, Y., AND IGARASHI, T. 2007. Plushie: an interactive design system for plush toys. *ACM Trans. Graph. 26*, 3 (Aug.), 45:1–45:7.

OTSU, N. 1979. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics 9*, 1, 62–66.

PRIYANTHA, N. B., CHAKRABORTY, A., AND BALAKRISHNAN, H. 2000. The Cricket location-support system. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, ACM Press, MobiCom '00, 32–43.

SAUL, G., LAU, M., MITANI, J., AND IGARASHI, T. 2011. SketchChair: an all-in-one chair design system for end users. In *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction*, ACM Press, TEI '11, 73–80.

SELLS, E., SMITH, Z., BAILARD, S., BOWYER, A., AND OLLIVER, V. 2009. RepRap: The Replicating Rapid Prototyper-maximizing customizability by breeding the means of production. *Handbook of Research in Mass Customization and Personalization 1*, 568–580.

SHIRATORI, T., PARK, H. S., SIGAL, L., SHEIKH, Y., AND HODGINS, J. K. 2011. Motion capture from body-mounted cameras. *ACM Trans. Graph. 30*, 4 (July), 31:1–31:10.

SHOPBOT TOOLS. ShopBot. `http://www.shopbottools.com/`.

SZELISKI, R. 2006. Image alignment and stitching: a tutorial. *Found. Trends. Comput. Graph. Vis. 2*, 1 (Jan.), 1–104.

TECHNO CNC ROUTER SYSTEMS. TechnoVision.

TRIGGS, B., MCLAUCHLAN, P. F., HARTLEY, R. I., AND FITZGIBBON, A. W. 2000. Bundle adjustment - A modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, Springer-Verlag, ICCV '99, 298–372.

WELCH, G., AND FOXLIN, E. 2002. Motion tracking: no silver bullet, but a respectable arsenal. *IEEE Comput. Graph. Appl. 22*, 6 (Nov.), 24–38.

WEYRICH, T., DENG, J., BARNES, C., RUSINKIEWICZ, S., AND FINKELSTEIN, A. 2007. Digital bas-relief from 3D scenes. *ACM Trans. Graph. 26*, 3 (Aug.), 32:1–32:7.

XIN, S., LAI, C.-F., FU, C.-W., WONG, T.-T., HE, Y., AND COHEN-OR, D. 2011. Making burr puzzles from 3D models. *ACM Trans. Graph. 30*, 4 (July), 97:1–97:8.

ZAHNERT, M. G., FONSEKA, E., AND ILIC, A., 2010. Handheld Scanner with High Image Quality. U.S. Patent US 2010/0296140 A1.

ZITOVA, B., AND FLUSSER, J. 2003. Image registration methods: a survey. *Image and Vision Computing 21*, 11 (Oct.), 977–1000.