Proceedings of the ASME 2010 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2010
August 15-18, 2010, Montreal, Quebec, Canada

DETC2010-28227

TRAJECTORY CONTROL FOR AN INNOVATIVE RAPID FREEZE PROTOTYPING SYSTEM

Eric Barnett * **&**
Jorge Angeles &
Damiano Pasini
Department of Mechanical Engineering
McGill University
817 Sherbrooke Street West
Montreal,QC H3A 2K6
Email: ebarnett@cim.mcgill.ca, angeles@cim.mcgill.ca
damiano.pasini@mcgill.ca

Pieter Sijpkes
School of Architecture
McGill University
815 Sherbrooke Street West
Montreal,QC H3A 2K6
Email: pieter.sijpkes@mcgill.ca

ABSTRACT

The subject of this paper is trajectory control for an Adept Cobra 600 robot, which has been retrofitted for additive ice construction. Since this application is quite different from typical SCARA applications, considerable development is needed for trajectory control and data flow. We first outline the trajectory control system requirements, as well as the limitations of the robot hardware. Different control options are proposed and their merits and demerits are discussed; the most suitable scheme is judged to be custom programming in Adept's V+ programming language, with trajectories expressed in the Cobra 600 joint space. With this control scheme, all system requirements are met, data manipulation is most efficient, and the system is readily adaptable for planned modifications. A specific data format is needed in order to implement this control scheme; we describe how trajectory data produced with our part-slicing algorithm is converted to the format required for the V+ programs.

INTRODUCTION

Ice construction has fascinated people for thousands of years. On the practical side, ice roads and shelters have enabled access to and settlement of remote areas. Artistic ice construction has a long tradition of its own; in recent decades, it has become more popular than ever. As new technologies have become available, ice construction is becoming increasingly automated. Computer numerical control (CNC) ice construction is quite well established, with several companies offering the capability to build certain parts on demand[1].

In traditional CNC machining, a part is formed by removing material using techniques such as milling and drilling. An obvious alternative to *material removal* is *material addition*, also known as rapid prototyping (RP) [1]. RP is a relatively new technology, first achieving widespread use in the 1980s. An important distinction exists between RP *deposition* systems such as fused deposition modeling (FDM) and systems such as selective laser sintering (SLS) and stereolithography, which selectively fuse material already in place. In the field of ice construction, Bryant and Leu have developed a deposition rapid freeze prototyping (RFP) system consisting of a valve/nozzle water delivery system positioned by stepper-motor driven axes [2–4].

There are significant advantages to using RP as opposed to traditional CNC techniques. Extremely complex parts can be built using RP that would either be impossible or require spe-

*Address all correspondence to this author.

[1] Ice Sculptures Ltd., Grand Rapids, MI (www.iceculture.com)
Ice Culture Inc., Hensall, ON (www.iceguru.com)

cific, expensive tooling using traditional CNC techniques. Also, the process from design to fabrication is much simpler using RP; it can be as simple as "printing" a 3D part much as one would print a 2D document with a regular printer. Of course, RP also has many drawbacks, the most obvious being fabrication time: a part that measures roughly 100 mm in each dimension could easily take 50 hours to build. Also, RP machines typically cost several hundred thousand dollars, and most build materials cost about $50/kg. Additional advantages of RFP construction include the use of an inexpensive, more environmentally-friendly build material and lower equipment costs, since a valve/nozzle deposition system is used and fume control infrastructure is not needed.

In [5], we reported on the long history of ice construction research at McGill University. In the past few years, we have focused on the development of computer-assisted ice construction techniques. At the small scale, two systems have been retrofitted for rapid prototyping with ice: an Adept Cobra 600 robot and a Fab@home desktop 3D printer [6]. For the Cobra 600 system, only the end-effector and part of the distal link are inside the freezer during construction. Our Cobra 600 RFP system is novel because the Cobra 600 has never before been used for RP, not to mention RFP. The 4-axis SCARA joint architecture is well-suited to RP, although many additional subsystems and considerable software development are needed to retrofit the system for RFP.

The RFP production process begins with a stereolithography (STL) part file, which has been generated from a CAD program or obtained from some other source. We have developed a Matlab code, which slices an STL file and generates the control data for building the part with our Cobra 600 RFP system [7]. Control data includes all trajectories the Cobra will follow to build the part, as well as valve timing data needed to control the deposition of water and the support structure through micro-solenoid valves and nozzles. Fluid is supplied under pressure from pressurized dispensing tanks outside the freezer; inside the freezer, the fluid lines are heated using an on/off temperature controller; the set point for the temperature near the nozzle tips is 20°C.

In this paper, we describe the trajectory control scheme developed for the Cobra 600 RFP system. Trajectory control is an important consideration in any RP application. For deposition RP systems, accurate control of the positioning speed is needed to ensure a uniform and accurate product. The nature of the trajectory control subsystem is influenced by the desired performance and the hardware and software limitations. It is desirable to have a *variable* positioning speed during deposition, since time can be saved if deposition is allowed to occur during tangential accelerations [8]. Additionally, since any RP machine will have a limit on the acceleration it can produce, paths with low curvature can be tracked much faster than paths with high curvature.

The deposition material flow rate must be proportional to the positioning speed to achieve a uniform cross-section of deposition. This can be achieved by: (a) holding both the material flow rate and the positioning speed constant during deposition; or (b) adjusting the material flow rate in real time when the tool accelerates. For FDM, neither of these options is used: material flow rate is kept constant and positioning speed varies considerably during deposition [9]. Boundary contours are typically filled in using the "cross-hatch" method, whereby several sharp 180 degree turns are followed during deposition, resulting in large accelerations and *overfill* and *underfill* errors [10, 11]. Various error-correction and compensation schemes have been developed to address this problem.

While FDM is the most widely-used and commercialized RP system, other deposition techniques offer more promise for variable flow control. For example, with the Fab@Home desktop rapid prototyping system, the deposition mechanism is a motor-driven syringe, controlled by a transistor-transistor-logic (TTL) signal [12]; in ballistic particle manufacture, a piezoelectric element controls flow by ejecting thousands of discrete drops per second [13]. Both of these systems, as well as our RFP systems, are candidates for variable flow control.

In our systems, the material flow rate is controlled by: (a) the pressure in the dispensing tank; (b) the TTL signal that controls the solenoid valve; and (c) the system temperature. Variation of any of these variables produces a nonlinear response in the flow rate. Initial tests have been conducted using the TTL signal for implementation of a variable flow control scheme. However, a better characterization of the nonlinear response produced in the flow rate is needed; additionally, we have observed that the flow stream produced is much less uniform at low flow rates. Therefore, to ensure a uniform and reliable flow, we must impose that the flow rate is not varied during deposition. In order to achieve a maximum part accuracy, this means that *deposition should only occur when the positioning speed is constant*; this constraint is also a novel feature of our RP system.

TRAJECTORY CONTROL REQUIREMENTS AND SYSTEM CONSTRAINTS

Requirements

In [7], we described a slicing algorithm developed to compute deposition paths for the Cobra 600, using an STL part file as input. Figure 1 shows an STL model of a statue of James McGill; a graphical representation of one layer of data output using the slicing algorithm is shown in Fig. 2. The trajectory data output from the slicing algorithm must undergo considerable postprocessing to satisfy the following requirements for optimal control of the Cobra 600:

1. Robot position, velocity, acceleration, and jerk must be controlled at all trajectory points

2. Data flow must accommodate a typical part composed of several million trajectory points
3. Trajectories must be adjustable to correct the measured errors in part accuracy
4. The system must be automated enough to be considered a "3D printer"

Requirement 1 results from the need to maintain constant speed along deposition paths. To ensure the smoothest paths possible, acceleration and jerk must be zero while deposition occurs. The non-deposition segments for acceleration and deceleration must also be controlled carefully, to respect the acceleration and jerk constraints along the deposition paths; additionally, correct valve signal timing is dependent on precise control of robot position at every instant.

Acceleration must also be limited to 600 mm/s^2 to prevent deflection of the end effector. Inertial forces can cause significant horizontal deflections of the EE because the vertical stroke of the Cobra 600 is 210 mm and the EE is 300 mm long, resulting in a 510 mm cantilever when the prismatic joint is fully extended. Three steps have been taken to limit EE acceleration: (a) the duration of non-depositing acceleration paths is set at 0.32 s; (b) the minimum point spacing is set at 0.8 mm; and (c) paths are broken at points where the direction of travel changes by more than 30°. Velocity, acceleration, and jerk constraints are also set to zero at path start and endpoints, since sharp changes in direction are often required when transitioning from one path to the next; additionally, this greatly simplifies valve control.

System Constraints

The Cobra 600 RFP system carries an Adept C40 Compact Controller with an AWC-II 040 Processor (25 MHz), 32 MB RAM, and a 128 MB CompactFlash disk. The Adept V+ operating system, which operates in a similar manner to DOS, is installed on the controller. Any PC can be used as a terminal for the controller, communicating with it through TCP/IP, RS232, or RS485. Additionally, we have the Adept AIM environment with the Pathware module installed: AIM is a graphical user interface designed to simplify programming tasks and development time; Pathware is intended specifically for path control applications such as material dispensing.

The most important decision to be made in the development of the trajectory control system is whether or not to use the AIM environment. The main advantage of using AIM is that many applications for Adept robots can be developed with little or no need for low-level V+ programming, resulting in reduced development time. Pathware provides additional controls for I/O timing, precise tracking, and speed control. Based on these characteristics, the AIM environment with the Pathware module should be an ideal environment for the development of our RFP trajectory control.

FIGURE 1. JAMES MCGILL STATUE STL MODEL

Trajectory points can be imported into AIM from any input file formatted in the AIM Database exchange (ADX), located on the local disk or in shared network folders. In this format, approximately 50 fields must be filled for every trajectory point. In practice, we have found that less than 10 points per second can be imported using this method; additionally, the system becomes unacceptably slow if more than 1000 points are stored in memory at once. We were able to overcome the total points limitation by writing a custom STATEMENT in AIM. However, only minimal improvements to the point importation speed were possible. Considering this restriction and other drawbacks such as the use of significant memory and processing resources, the AIM environment offers little advantage over customized V+ programming.

CUSTOMIZED V+ DEPOSITION PROGRAM

Custom V+ programming allows for the greatest amount of freedom during application development. However, it also increases development time, since a reasonable command of the V+ language is needed.

The three largest advantages to developing custom V+ programs are:

1. Flexibility in data storage and processing
2. Availability of robot control in joint space, if needed

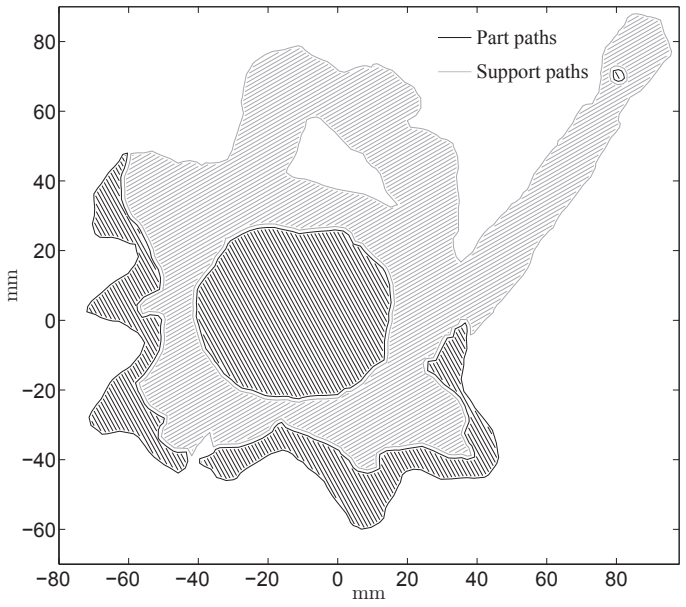

FIGURE 2. PART AND SUPPORT STRUCTURE DEPOSITION PATHS FOR ONE SLICE OF THE JAMES MCGILL STATUE STL MODEL

3. Availability of additional commands for the development of smooth trajectory profiles

In V+, a simple READ command is used to access data stored in a file; standard formats such as *character*, *integer*, and *float*, are available. Data can be read into multidimensional arrays stored in the controller memory; the maximum length along any dimension of an array being 32,768. For our application, only four variables need to be imported for each trajectory point: the three joint angles and the time of travel from one point to the next. This means that the required array size for a set of trajectories is $4 \times n$ in size, where n is total number of trajectory points to be imported. Trajectory data files are stored in the IEEE single-precision floating-point format on a network PC: a shared folder on the PC's hard drive is configured as a network drive using the Omni-NFS Server software. Since 16 bytes are required to store each trajectory point, a part with several million points can require several hundred MB of storage space. As the memory and storage space is limited on the Cobra controller, the best option is to import trajectory data one layer at a time and overwrite the data once it has been used. When this data scheme is implemented, a significant increase in efficiency is observed over the AIM environment: trajectory points can be imported at a rate of approximately 2000 per second.

Cobra 600 Coordinate Systems

When importing and following trajectory data using the V+ programming language, one can choose to operate in Cartesian space or in joint space. The Cobra 600 is an RRPR manipulator, where R denotes a revolute joint and P denotes a prismatic joint. In Cartesian space, trajectory points must be stored in the format (*x*, *y*, *z*, *yaw*, *pitch*, *roll*) where *x*, *y*, and *z* specify the location, and *yaw*, *pitch*, and *roll* specify the orientation of the center point of the Cobra User Flange—the end-effector (EE) attachment point. Since the Cobra 600 is a four-axis manipulator, *yaw* and *pitch* are both constant and can be omitted; *roll* is the orientation of the User Flange in the *xy* plane.

We define the trajectory resolution r_t as the number of points per millimeter stored in memory for a trajectory to be followed. It is desirable for r_t to be as high as possible in order to maximize part accuracy. However, if r_t is too high, the Cobra controller becomes overloaded and the robot fails to follow the trajectory at the correct speed. This situation must be avoided since it leads to a significant loss of part accuracy. Based on observation, point spacing in a trajectory file can be as small as 2 ms, whether joint space or Cartesian space is used. This is an expected result, since the trajectory generation rate of the controller is set at 500 Hz. It is important to note that this result is valid only when the controller processor does not have any other heavy programming tasks and the end-effector is traveling below a certain speed. For example, a programmed point spacing of 2 ms will work correctly at a programmed speed of 50 mm/s, resulting in a trajectory resolution $r_t = 0.1$ mm. At a *programmed* speed of 100 mm/s, however, the *actual* speed will be less than 100 mm/s, and the *actual* point spacing will be more than 2 ms.

It is preferable to represent points in joint space, where each programmed point corresponds to a single robot posture; in Cartesian space, each programmed point can correspond to multiple robot postures. Additionally, trajectory importation speed in V+ is approximately 10% faster when points are represented in joint space.

Position, Velocity, Acceleration, and Jerk Control

One of the system requirements is to control position, velocity, acceleration, and jerk at all points on a trajectory. A conventional technique for trajectory control would be to use the V+ MOVE command along with the SPEED and ACCEL commands. However, as point resolution is increased, the SPEED command starts to become unreliable. Also, only square-wave and trapezoidal acceleration profiles can be configured using the ACCEL command. Therefore, this technique cannot be used to control position and its first three derivatives. With the DURATION command, however, a minimum duration of travel is specified for every trajectory segment: $n-1$ DURATION values need to be specified for n trajectory points. As long as a DURATION value is physically possible, it will correspond to the exact duration of travel. In this fashion, the time at which the end-effector passes through every trajectory point can be specified, providing control over robot position and position derivatives. Obviously,

it is desirable to have the maximum point resolution possible to minimize discretization effects.

Required Data Format and Planned Application Enhancements

We can thus conclude that if customized V+ programming is used, all trajectory control requirements can be met, as long as imported trajectory data adhere to the following format: locations are expressed in joint space; locations are spaced by at least 2 ms; non-depositing segments are added for acceleration at the start and end of each trajectory; and a DURATION value is specified at each point.

A major enhancement to the Cobra 600 RFP system will be the development of closed-loop control for deposition. In a planned *offline* feedback system, part height will be measured at several locations, height errors will be computed, and the error signals will be used to adjust trajectory control data in subsequent layers. Since deposition control parameters all affect flow rate in a nonlinear manner, we plan to instead adjust the DURATION values, which will affect EE speed linearly. Online control will be considered if there is time available to implement it and if we believe it can offer significant improvements over the offline scheme. One drawback of an online scheme is the need to program EE orientation so that both the deposition tool and the measuring laser beam strike the trajectory being followed. Also, care will need to be taken to avoid over-rotation of joint 4, which is only capable of two complete revolutions. Another drawback of online control is a significant increase in the real-time processing load for the Cobra controller, which could require sacrifices in trajectory accuracy.

TRAJECTORY DATA CONVERSION

The main postprocessing steps needed to transfer the data output from the slicing algorithm are: choice of a suitable trajectory model; redistribution of trajectory points; addition of non-depositing paths; and transfer of points from Cartesian to joint space by solving the inverse displacement problem (IDP). All algorithms described in this section are programmed with Matlab R2009b, unless otherwise indicated.

Trajectory Model

As described above, the trajectory data required by the Cobra are of a discrete nature. However, since velocity, acceleration, and jerk are all time derivatives of position, it is reasonable to start from a continuous trajectory model, which will then be discretized at the necessary resolution. We will produce position, velocity, acceleration and jerk using *normalized* scalar functions $s(\tau)$, $s'(\tau)$, $s''(\tau)$, and $s'''(\tau)$ in non-dimensional time τ.

In order to minimize positioning errors, system vibration, and abrupt motion, it is desirable to prescribe smooth trajectories: acceleration and jerk should vanish at the endpoints. In [14, p. 241], a 4-5-6-7 polynomial is used to synthesize such a trajectory, which also has zero velocity at the endpoints. For the RFP system, the objective is to reach the deposition speed $v_{\max}$ as smoothly as possible, while respecting the maximum acceleration the system can be subjected to. Therefore, non-depositing path segments, synthesized with 4-5-6-7 polynomials, are attached to the endpoints of each constant-speed deposition path. Each of these polynomials has the form:

$$s(\tau) = a\tau^7 + b\tau^6 + c\tau^5 + d\tau^4 \tag{1}$$

When the end conditions for the acceleration segment are imposed, the following solution is found:

$$a = 0, \quad b = 1, \quad c = -3, \quad d = 2.5$$

which means that $s(\tau)$ is actually a 4-5-6 polynomial in this case. The maxima of $s(\tau)$ and its first three derivatives turn out to be

$$s_{\max} = 0.5, \quad s'_{\max} = 1, \quad s''_{\max} = 15/8, \quad s'''_{\max} = 10\sqrt{3}/3$$

The non-dimensional 4-5-6-7 polynomials are used to produce dimensioned trajectories, subject to required constraints. Figure 3 shows a trajectory synthesized with $v_{\max} = 100$ mm/s, an acceleration time of 0.32 s, and path length 100 mm.

Experimental Results

The path shown in Fig. 3 is discretized with point-to-point intervals of 16 ms; since $v_{\max} = 100$ mm/s, this means trajectory points are 1.6 mm apart along the constant speed segment. Closer point spacing could not be used in this case, because significant controller resources were needed to record the position data from the encoders. Plots of the prescribed trajectories, compared to those of the actual trajectory followed by the Cobra 600, are shown in Fig. 4.

The actual trajectory curves were generated from filtered encoder position data, sampled at 1000 Hz. These data tend to be extremely noisy, indicating that strong filtering is needed. However, for this application, the filtering can be done *offline*, since the data are not being used for real-time control. This distinction is important, since most filters described in the literature are designed for control systems. In light of these factors, we chose to implement a least squares fit (LSF) filter similar to that described by Carpenter et al. [15]. For this filter, a polynomial fitting curve is computed for *every* position reading, based on that reading

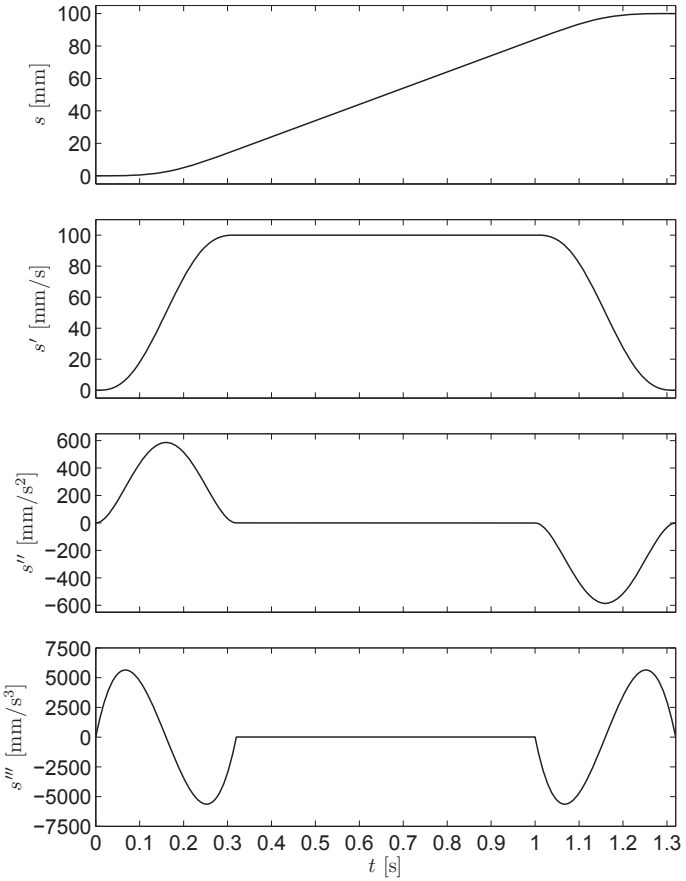

FIGURE 3. A TRAJECTORY WITH A 4-5-6-7 POLYNOMIAL SEGMENT FOR ACCELERATION, A STRAIGHT-LINE SEGMENT FOR COASTING, AND A SECOND 4-5-6-7 POLYNOMIAL SEGMENT FOR DECELERATION

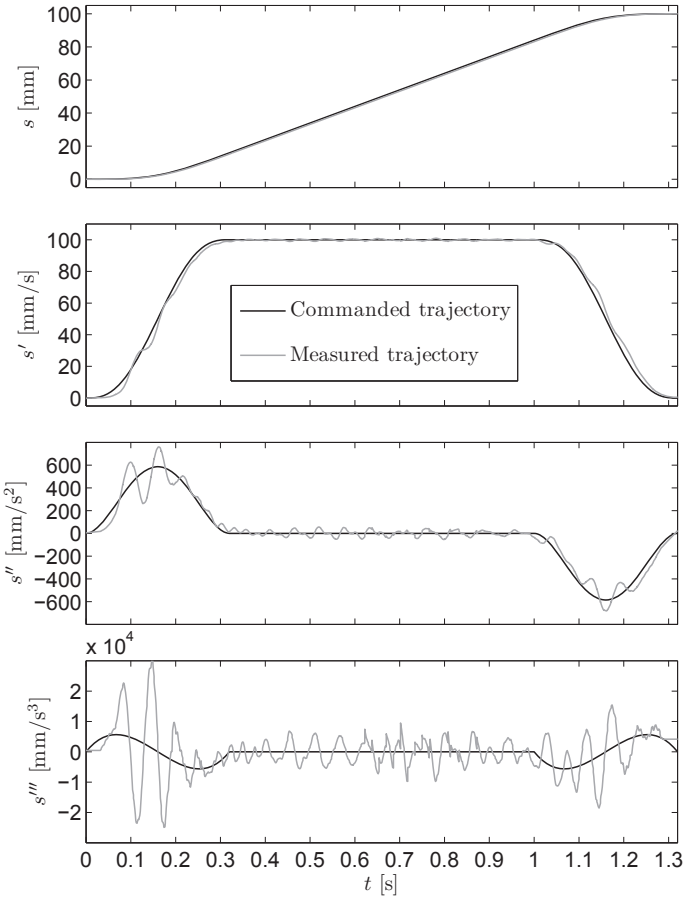

FIGURE 4. A COMPARISON THE TRAJECTORY SYNTHESIZED WITH 4-5-6-7 POLYNOMIALS AND THE ACTUAL TRAJECTORY FOLLOWED BY THE COBRA 600

and the *m* previously recorded readings. This curve is evaluated at the time value of the unfiltered position to obtain the filtered position.

The most important parameters that must be chosen for the LSF filter are the polynomial order for the fitting curve and the value for *m*. In order to have a jerk estimate, the polynomial must be of at least third order, though it should also be as low as possible to avoid noise interpolation. The value of *m* should be high enough to suppress noise-related plot characteristics. For example, significant negative accelerations observed while the end effector speed is clearly increasing indicates the need for stronger filtering. Filtering is too strong if non-realistic plot characteristics, such as nonzero initial velocity, are observed.

Since the LSF filter is implemented *offline*, encoder data before and after the location to be filtered can be used. We found that a 64-point LSF filter of the third order resulted in the most reasonable filtering of encoder data. 32 points before and 32 points after each encoder point were used to calculate the third order polynomial. For the first 32 and last 32 points of the trajectory, the LSF polynomials for the 32nd and 32nd-to-last points, respectively, are used. In this way, a third-order LSF polynomial is computed for every recorded encoder point. These polynomials are evaluated to obtain filtered position data; velocity, acceleration, and jerk are found by computing and evaluating polynomial derivatives.

Figure 4 shows a comparison of the trajectory synthesized with 4-5-6-7 polynomials and the actual trajectory followed when measured using the technique described above. It is difficult to be certain whether the oscillations visible at the velocity level and amplified at the acceleration and jerk levels can be ascribed to noise amplification or actual position errors. From these plots, we can conclude that velocity and acceleration are very close to zero at the start and end of the trajectory. Little can be concluded from the jerk plot. More accurate estimates of acceleration and jerk are planned through the use of an accelerometer attached to the end effector.

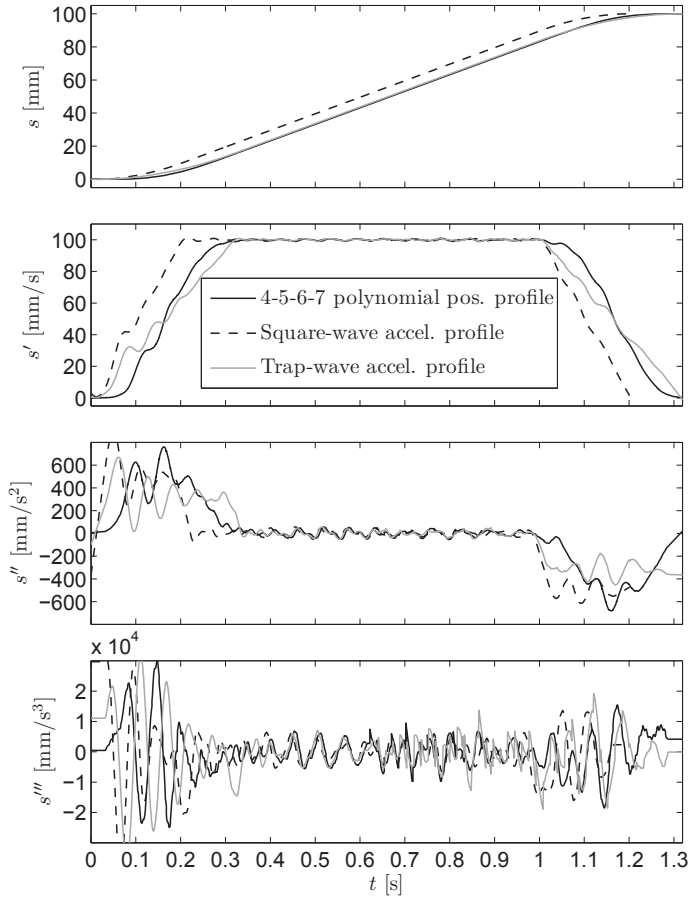

FIGURE 5. A COMPARISON OF FOLLOWED PATHS SYNTHESIZED WITH ACCELERATION PROFILES AVAILABLE IN V+ AND A FOLLOWED PATH SYNTHESIZED WITH 4-5-6-7 POLYNOMIALS.

Figure 5 shows a comparison of paths synthesized with acceleration profiles available in V+ and the path synthesized with 4-5-6-7 polynomials. For one of these paths, a square-wave acceleration profile is used with the default acceleration time of approximately 0.2 s; for the other, a trapezoidal profile is used, with a 0.107 s rise time and an acceleration time of 62% longer than the minimum time. The most important features to recognize from these plots are that the velocity slope and the acceleration both vanish at the path endpoints for the the path synthesized with 4-5-6-7 polynomials; this does not occur for other paths.

In addition to the data analysis described above, a qualitative comparison of the trajectories was performed. A wire measuring approximately 300 mm long was attached to the end effector and bent in the vertical direction. When following the square-wave and trapezoidal acceleration profiles, the wire oscillated noticeably for several seconds after the robot came to rest; using the 4-5-6-7 polynomial trajectory, the magnitude and duration of this oscillation were both much smaller.

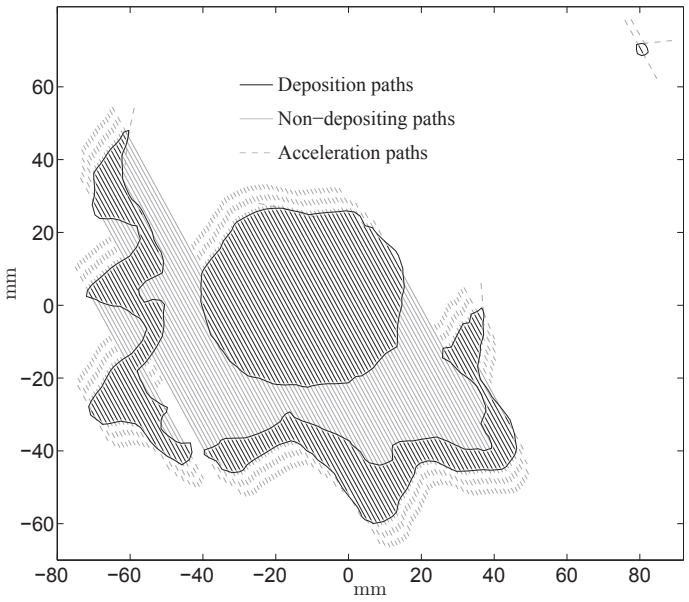

FIGURE 6. DEPOSITION, NON-DEPOSITING, AND ACCELERATION PATHS FOR ONE SLICE OF THE JAMES MCGILL STATUE STL MODEL

Redistribution of Trajectory Points Output From the Slicing Algorithm

Position data shown graphically in Fig. 2 must be modified to conform to the 4-5-6-7 polynomial trajectories described in the previous section. Since it is desirable for deposition to occur only when the end-effector is moving at $v_{\max}$, acceleration and deceleration paths are added to the deposition paths by computing the tangents to the start- and end-segments. This is shown graphically in Fig. 6, where support paths are not shown because they would overlap the other lines. As seen in Fig. 6, there are often multiple deposition segments on the same line. Rather than accelerate and decelerate for each segment, constant speed is maintained and deposition is turned on and off as required; this can be done accurately because of the precise timing achievable when using the DURATION command.

Inverse Displacement Problem (IDP)

Once the path data are in the format shown graphically in Fig. 6, trajectory points are converted from Cartesian to joint space. The Cobra 600 end-effector (EE) geometry is shown in Fig. 7. The joint angles θ_i that place the operation point P of the EE at position (x_P, y_P) and orientation ϕ, need be found. The IDP for the Cobra 600 may be solved using the same procedure as that for a 3R planar manipulator [14, p. 193], which is standard practice and need not be included here.

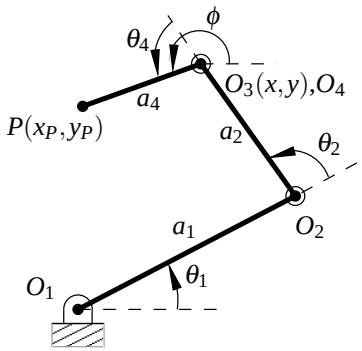

FIGURE 7. THE COBRA 600 RRPR MANIPULATOR GEOMETRY

There are two real solutions for θ_1, corresponding to "lefty" and "righty" configurations of the Cobra 600. For our application, the smaller solution, corresponding to the "righty" configuration, is always used to prevent interference with other equipment in the workcell. The final joint angle, θ_4, is found from the geometry of Fig. 7. Joint variable b_3, associated with the P joint, is not needed in this analysis.

We define P as *the location where the liquid jet strikes the substrate or the part being built.* A more obvious choice for P would be the exit point of the nozzle, though if the nozzle is not perfectly aligned with the robot z direction, or if the stream is not perfectly aligned with the nozzle, this definition will result in part error. It is thus difficult to obtain an accurate measurement for a_4; also, stream geometry has been observed to vary when flow rate control parameters are changed. Given these system characteristics, it is desirable to avoid defining a_4 during the solution of the IDP. If ϕ is held constant, a_4 need not be specified, since vector $\overrightarrow{O_3P}$ will be constant. In this way, the exact position of the ice part being built will be unknown in the Cobra 600 base coordinate frame; however, the velocity between points will be more accurate because the relative position of different (x,y) coordinates will not depend on a_4.

If a precise measurement of a_4 could be made, ϕ could be used to optimize the system in various ways. For example, ϕ could be specified to obtain optimal conditioning of the Jacobian matrix; alternatively, ϕ could be used to position the laser measurement system over the deposition paths in an online deposition feedback system. However, an important limitation on the end-effector orientation is that the operable range for θ_4 is 720°. If, during the course of a commanded path, an impossible θ_4 value is input, a rapid de-spinning of Joint 4 can occur.

While the IDP is quite simple for the Cobra 600, it is very important to ensure that it is computed *efficiently*, since several million points will be processed for a typical part. Maple is used to compute the symbolic solution, leaving x and y as variables. This solution is expressed explicitly in Matlab and applied to Cartesian points *without* the use of any `solve` commands.

FIGURE 8. JAMES MCGILL STATUE IN ICE: 760 LAYERS, TOTAL BUILD TIME: 32 HOURS.

CONCLUSIONS

The trajectory control scheme outlined in this paper bridges the gap between STL model-slicing and deposition for the Cobra 600 RFP system. This increases system automation and brings it closer to something that can truly be regarded as a 3D printer. We will now conduct a period of testing and validation by building various types of physical models with the system. The trajectory control scheme is also highly adaptable to future modifications. Since data are loaded *incrementally*, the system can easily be adapted to accommodate the construction of multiple parts at once. Additionally, feedback deposition control is planned through the measurement of part height and modification of the DURATION values in the trajectory data for subsequent layers.

A manufactured prototype of the James McGill ice statue can be seen in Fig. 8; the model is approximately 150 mm tall, consists of 760 layers, and required 32 hours of total construction time.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of The Social Sciences and Humanities Research Council of Canada (SSHRC), the Natural Sciences and Engineering Research Council of Canada (NSERC), the Fonds québecois de la recherche sur la nature et les technologies (FQRNT), and McGill University's Faculty of Engineering. The generous rebate received from Adept Technology is dutifully acknowledged.

REFERENCES

[1] Crawford, R. H., and Beaman, J. J., 1999. "Solid freeform fabrication". *IEEE Spectrum,* ***36***(2), pp. 34–43.

[2] Zhang, W., Leu, M. C., Yi, Z., and Yan, Y., 1999. "Rapid freezing prototyping with water". *IEEE Spectrum,* ***20***, pp. 139–145.

[3] Bryant, F. D., Sui, G., and Leu, M. C., 2003. "A study on the effects of process parameters in rapid freeze prototyping". *Rapid Prototyping Journal,* ***9***(1), pp. 19–23.

[4] Bryant, F. D., and Leu, M. C., 2009. "Modeling and experimental results of concentration with support material in rapid freeze prototyping". *Rapid Prototyping Journal,* ***55***(1), pp. 317–324.

[5] Sijpkes, P., Barnett, E., Angeles, J., and Pasini, D., 2009. "The architecture of phase change at McGill". *Architecture Research Centers Consortium Spring Conf. (ARCC 2009)*, San Antonio, TX, April 15–18, 2009, 6 pages.

[6] Barnett, E., Angeles, J., Pasini, D., and Sijpkes, P., 2009. "Robot-assisted rapid prototyping for ice structures". *IEEE Int. Conf. on Robotics and Automation*, Kobe, Japan, May 12–17, 2009, pp. 146–151.

[7] Ossino, A., Barnett, E., Angeles, J., Pasini, D., and Sijpkes, P., 2009. "Path planning for robot-assisted rapid prototyping of ice structures". *Trans. Can. Soc. Mech. Eng.,* ***33***(4), pp. 689–700.

[8] Fadel, G., and Ganti, R., 1998. "Parametric based controller for rapid prototyping applications". *Proc. of the Solid Freeform Fabrication Symposium*, Austin, TX, Aug. 10–12, 1998, pp. 236–243.

[9] Bouhal, A., Jafari, M. A., Han, W. B., and Fang, T., 1999. "Tracking control and trajectory planning in layered manufacturing applications". *IEEE Trans. on Industrial Electronics,* ***46***(2), pp. 445–451.

[10] Han, W., Jafari, M. A., Danforth, S. C., and Safari, A., 2002. "Tool path-based deposition planning in fused deposition processes". *Trans. of the IEEE,* ***124***, pp. 462–472.

[11] Lu, X., Lee, Y., Yang, S., Hao, Y., Evans, J. R. G., and Parini, C. G., 2009. "Fine lattice structures fabricated by extrusion freeforming: Process variables". *J. Materials Processing Technology,* ***209***, pp. 4654–4661.

[12] Malone, E., and Lipson, H., 2007. "Fab@home: The personal desktop fabricator kit". *Rapid Prototyping Journal,* ***13***(4), pp. 245–255.

[13] Wallace, D. B., Cox, W. R., and Hayes, D. J., 2002. *Direct Write for Ink-Jet Techniques*. Academic Press, New York, ch. 7.

[14] Angeles, J., 2008. *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*, third ed. Springer-Verlag, New York.

[15] Carpenter, P. S., Brown, R. H., Heinen, J. A., and Schneider, S. C., 1995. "On algorithms for velocity estimation using discrete position encoders". *IEEE IECON 21st Int. Conf. on Industrial Electronics, Control, and Instrumentation*, Orlando, FL, Nov. 6–10, 1995, ***2***, pp. 844–849.