# Distortion-free Robotic Surface-drawing using Conformal Mapping

Daeun Song and Young J. Kim

*Abstract*— **We present a robotic pen-drawing system that is capable of faithfully reproducing pen art on an unknown surface. Our robotic system relies on an industrial, seven-degree-of-freedom manipulator that can be both position- and impedance-controlled. In order to estimate a rough geometry of the target, continuous surface, we first generate a point cloud of the surface using an RGB-D camera, which is filtered to remove outliers and calibrated to the physical canvas surface. Then, our control algorithm physically reproduces digital drawing on the surface by impedance-controlling the manipulator. Our impedance-controlled drawing algorithm compensates for the uncertainty and incompleteness inherent to a point-cloud estimation of the drawing surface. Moreover, since drawing 2D vector pen art on a 3D surface requires surface parameterization that does not destroy the original 2D drawing, we rely on the least squares conformal mapping. Specifically, the conformal map reduces angle distortion during surface parameterization. As a result, our system can create distortion-free and complicated pen drawings on general surfaces with many unpredictable bumps robustly and faithfully.**

## I. INTRODUCTION

In the early twentieth century, contemporary art challenged conventional approaches by experimenting and exploring new techniques. Since then, numerous new attempts have emerged to fuse science and technology into an art form. Along the way, mechanical machines also have become means and subjects of these art forms adopted by a few artists [1],[2],[3],[4]. Robots are now operating in our daily lives and become a subject of art. Due to the recent impressive development of robotic technology, the artistic roles of robots become more diverse, and possibilities to express artists' intention through robotic mechanism and interaction with it is expanding [5].

In this paper, we present our new research work on a robotic pen-drawing system that relies on human's creativity but produces a new creation of artwork. In our previous work [6], we introduced an artistic robotic drawing system that can create visually-pleasing and complicated artistic pen drawings on general surfaces without explicit surface-reconstruction nor visual feedback. In order to expand our system to be scalable in terms of canvas size and drawing time, now we employ an RGB-D camera to estimate the shape of the canvas surface geometry.

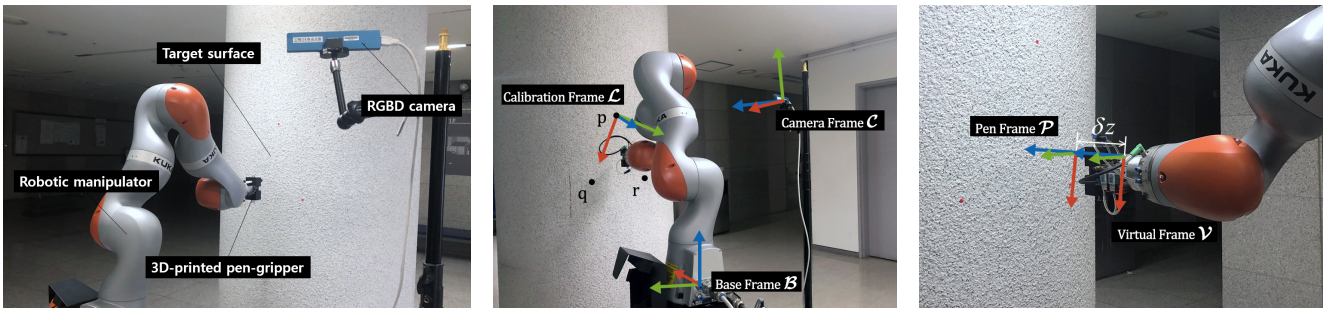
Simply adding the vision capability to our system, however, may not only complicate the drawing pipeline but also even worsen the robotic drawing results, as the sensor would introduce different sources of noises from depth estimation and calibration. Moreover, in our previous work, we ignored a mapping artifact from a 2D vector graphic drawing to a 3D robotic drawing, that can introduce serious image distortion for an arbitrary, bumpy surface.

Robotic surface drawing without inducing bad visual artifacts is not just limited to artistic drawing, but also applicable to more practical tasks. For instance, in a robotic cleaning application, a robot needs to follow a trajectory (or a cleaning pattern) while making contact with the target cleaning surface [7]. In this case, the cleaning pattern is pre-determined in 2D space (*e.g.,* a space-filling curve in 2D), and the cleaning surface will not be known a priori and can be also defined in 3D (*e.g.,* a window-cleaning robot). The task objective here would be to sweep the 3D surface as closely following the given patterned trajectory as it would on the 2D surface - *i.e.,* the pre-determined cleaning pattern in 2D should be faithfully reproduced in 3D without distorting the patterns. A similar scenario is possible for a surface-probing robot where the robot needs to evaluate and reconstruct an unknown surface by following a pre-determined trajectory, also known as manifold learning [8], [9].

**Main Results:** In contrast to our previous work [6], now we use minimal vision support using an RGB-D camera to estimate the unknown target canvas-surface and represent it as a point cloud. However, since this estimation is often too noisy for an arbitrarily bumpy surface to draw pen art on it, we rely on an impedance-control technique to counteract the potential uncertainties caused by the sensor noise as well as the numerical noise in surface estimation. Moreover, in order to map the original vector drawing in 2D to a general surface in 3D while minimizing mapping distortion, we use an idea of conformal mapping to mitigate the distortion. Since conventional conformal mapping techniques developed in raster-based computer graphics are not directly applicable to our problem, as our input drawing is not rasterized but piecewise continuous curves, we employ bi-linear interpolation to compute proper conformal parameterization. Finally, we successfully demonstrate that our robotic drawing system can generate visually-pleasing pen drawings on real-world surfaces with minimal distortion.

The rest of this paper is organized as follows. We survey works relevant to robotic surface drawing in Sec. II. We propose our surface estimation method in Sec. IV and the robotic surface drawing method with minimal distortion in Sec. V. We explain the robotic incarnation of this technique using an impedance-control technique in Sec. VI. We show our implementation results and discuss them in Sec. VII, and conclude the paper in Sec. VIII.

The authors are with the department of computer science and engineering at Ewha womans university in Korea daeun7250@ewhain.net,kimy@ewha.ac.kr




(a) Robotic setup (b) Coordinate system setup (c) Tool frame setup

Fig. 1. Robotic Surface-drawing System Setup

## II. Previous Work

### A. Robotic Surface Drawing

A good application of robotic surface drawing is an artistic robotic drawing. Tinguely and Aaron [10] pioneered this area by creating a drawing machine in the 1990s. An earlier prototype of drawing robots is mostly based on a special-purpose plotter robot, and the recent trend is to use a high-DoF robotic manipulator for robotic drawing that can is more versatile in terms of artistic expressions.

A telerobotic work such as the Pumapaint project [11] allows remote users to draw paintings using a classical PUMA robot. The HOAP2 humanoid robot was also employed to draw human portraits in human-characteristic styles [2]. They also rely on speech recognition and synthesis to create an interactive system. Paul the robot [4] is a robotic installation that creates observational portrait drawing, mimicking an artist's stylistic signatures. The robot itself is a planar robotic arm using a pen as end-effector with a tilt webcam for visual feedback. This robot was also publicly displayed during the ICRA 2018 conference. eDavid [3] is a response to robotic art from the computer graphics community and can draw non-photorealistic paintings from an image input using an industrial robot. A drone-type robot was introduced to create stippling effects from an image input [12]. The drone is equipped with an ink-soaked sponge and uses a centroidal Voronoi diagram to generate stippling effects and minimizes its motion by approximating the traveling salesman problem. Song et al. [6] used a collaborate robot to draw artistic pen-drawing on an unknown surface using impedance-control, but ignored other sensory information including robotic vision and is unable to generate a distortion-free rendering of drawings on an unknown surface.

### B. Distortion-free Surface Parameterization

Automatic parameterization of a non-parametric surface like polygonal or point-set surfaces is a non-trivial problem and has been extensively studied in geometric modeling and processing [13], [14]. Typical applications of parameterization include texture mapping in computer graphics, and mesh editing and re-meshing in geometry processing. In particular, the former application is closely related to our problem, where a curved drawing in 2D should be faithfully reproduced on curved or bump surfaces in 3D.

For a polygonal surface with disk-like topology and a convex boundary, Barycentric mapping [15] is the most popular method in the literature. However, this method can induce serious distortion in parameterization if the boundary is highly non-convex. Coping with this problem, conformal mapping has been introduced based on a complex analysis. At a high level, conformal mapping can be classified into analytic and geometric methods [14]. The former is relatively easy to implement using energy minimization [16] but can suffer from unbalanced distortion if the underlying surface has high Gaussian curvature while the latter can resolve this issue [17].

### C. Impedance-controlled Robot

Hogan et al. [18] first suggested an idea of using impedance control for controlling the interaction between a manipulator and the surrounding environment. Since then, robot interaction techniques using impedance control have been explored in the robotics community [19], [20]. A popularity of the use of impedance control is partly due to the demand for robotic systems with an ability to interact with uncertain environments in practical applications. Thus, many collaborative robots based on impedance-control have been introduced from both industry and academia, for instance, LBR IIWA from KUKA Robotics, Baxter and Sawyer from Rethink Robotics, and Justin from DLR. These robots are able to handle classically challenging robotic manipulation such as peg-in-hole assembly. Impedance control can be also used for unscrewing a can for a humanoid robot Justin [21] as well as bimanual tasks [22]. The use of impedance control for drawing is very new, and Song et al.'s work [6] is the only known work to the best of our knowledge.

## III. System Overview

As illustrated in Fig. 1-(a), we show the setup of our robotic drawing system consisting of a robotic manipulator equipped with a 3D-printed pen-gripper to hold a pen, a non-planar target surface $S$ and an RGB-D camera. In Fig. 1-(b), we also show different frames used for system implementation where the robot base's frame $\mathcal{B}$ coincides with the inertial frame $\mathcal{W}$ in our system. The input drawing of our system consists of a sequence of control points $C$ in 2D, defining quadratic Bézier curves, which are drawn by an

artist using tablet-like pen interfaces such as [6]. To project this drawing on to the target surface $S$, we obtain a point cloud data $P$ representing the surface $S$ using an RGB-D camera. After calibrating $P$ with $S$ with respect to $\mathcal{W}$, we estimate the normal vector for each point in $P$, later used for conformal mapping as well as for orienting the robot's pen frame. Then, the surface $S$ is parameterized into 2D domain $\Omega$ by first approximating it using mesh reconstruction from $P$ and then performing conformal mapping on it.

The original input drawing defined by $C$ is made to fit in the domain of $\Omega$, and their corresponding parameters are found by inverse-mapping $\Omega$ to $P$ and performing bi-linear interpolation on it. Finally, the robot performs drawing with impedance control to compensate for possible mapping and estimation errors.

## IV. SURFACE ESTIMATION

In order to reproduce the drawing originally provided as a sequence of 2D vectors on a target surface, we first acquire depth information of the target surface. Various types of RGB-D cameras, such as stereo vision, time-of-flight, can reconstruct the depth information of a 3D surface with an accuracy range of $\pm 5\%$ with a resolution less than $1\text{mm}$.

Using this depth data, we generate a point cloud of the scene and calibrate it with the target surface by defining a local frame, called calibration frame, attached to the target surface using three non-degenerate points on it. Further, we also estimate the surface normals for the calibrated points to orient the robot's tool frame with respect to the robot base frame. This enables the robot's pen tip aligned toward the surface normal of the target surface and also to exert proper forces on to the pen tip to draw. However, since these estimated results may not be precise due to sensor noise, we use impedance-based control to compensate for the estimation error that will be explained in Sec. VI.

### A. Point Cloud Calibration

Calibrating the captured point cloud $P$ data with the source surface $S$ is crucial for distortion-free robotic drawing. However, common calibration methods based on planar markers or mounting a camera on the robot arm is not suitable for our purpose because our target surfaces are often non-planar, and they are placed very close to the robot manipulator that it violates the RGB-D camera's required minimum distance. So we devise our own approach as follows. With a captured point cloud ${}^{C}P$ relative to the camera frame $\mathcal{C}$, we transform it relative to the robot's base frame $\mathcal{B}$, which also corresponds to the inertial frame $\mathcal{W}$ in our approach.

To calibrate, we define a local calibration frame $\mathcal{L}$ attached to the surface $S$ using three non-degenerate position-markers $\{\mathbf{p}, \mathbf{q}, \mathbf{r}\}$ on the surface (Fig. 1-(b)). We define $\overrightarrow{\mathbf{pq}}$ as the x-axis and the normal vector $\mathbf{n}$ of the $\triangle\mathbf{pqr}$ as the z-axis of $\mathcal{L}$. By registering the position markers in the point cloud $P$, the frame $\mathcal{L}$ is defined with respect to $\mathcal{C}$. Similarly, by picking $\mathbf{p}, \mathbf{q}, \mathbf{r}$ using the robot's end-effector and using inverse kinematics, $\mathcal{L}$ with respect to $\mathcal{B}$ is redefined. Then, the relative transformations from $\mathcal{C}$ or $\mathcal{B}$ to $\mathcal{L}$ are defined respectively. Finally, ${}^{B}_{C}\mathbf{T}$, the transformation from $\mathcal{C}$ to $\mathcal{B}$ is obtained and applied to the point cloud data.

$$ {}^{B}_{C}\mathbf{T} = {}^{B}_{L}\mathbf{T}({}^{C}_{L}\mathbf{T})^{-1} \tag{1} $$

$$ {}^{B}P = {}^{B}_{C}\mathbf{T}\ {}^{C}P. \tag{2} $$

Fig. 2 illustrates a dense point cloud captured from a camera frame $\mathcal{C}$, transformed into the robot base frame $\mathcal{B}$.

### B. Normal Estimation

To perform robotic surface drawing on an arbitrary surface, the robot needs to decide both the position $\mathbf{p}$ and orientation $\mathbf{r}$ of its end-effector. In particular, to calculate the orientation $\mathbf{r}$ that ensures the robot to make stable contact on a surface point, we calculate the surface normals $\mathbf{n}$ of a point cloud and make it parallel to the approaching direction (*i.e.,* $z$-axis) of the end-effector (*i.e.,* pen).

One can consider two options to estimate the normals of a point set $P$. A rather straightforward way is to use a surface meshing technique to obtain a mesh representation of $P$ and then extract the per-vertex normals. Alternatively, one can infer per-point normal directly from $P$ by performing least-squared plane-fitting [23]. This is reduced to the principal component analysis for a local neighborhood of points. Even though either way would work in our case, the surface parameterization method we have chosen in Sec. V requires surface meshing, and thus opt for the meshing technique.

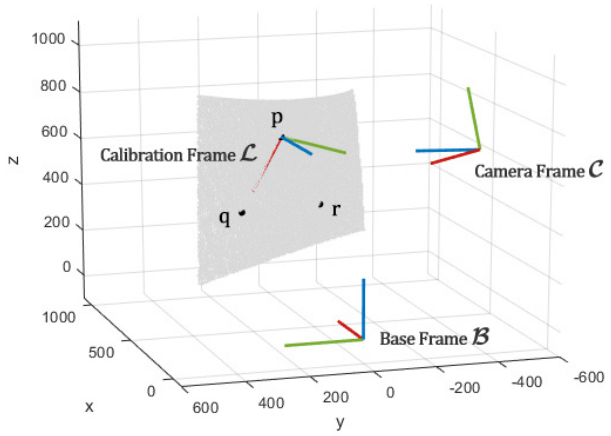



Fig. 2. Calibrated Dense Point Cloud

## V. DRAWING FROM 2D TO 3D

To reproduce a 2D drawing on an arbitrary surface in 3D without severe image distortion, one can employ conformal texture mapping. In other words, we can preserve the quality of the original 2D drawing during mapping from a 2D digital vector-drawing to a robotic drawing on a 3D surface by minimizing the angle distortion of local geometry. As a result, we can get a distortion-free drawing result regardless of whether the drawing is executed on a large or small, a stretched or shrunken surface. However, conventional conformal texture mapping methods used in rasterization-based computer graphics is not directly applicable to our problem

as our input drawing data is composed of a sequence of spline curves in 2D using a set of control points $C$, corresponding to pen strokes. Moreover, our target surface is a set of 3D points with estimated normals, but without explicit surface parameterization.

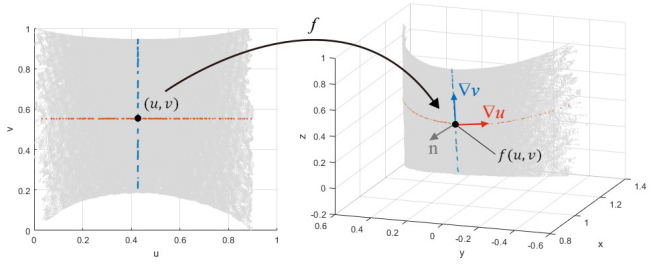



Fig. 3. Conformal Mapping

### A. Conformal Mapping

Before explaining our idea of conformal mapping, we first introduce some theoretical background of conformal mapping. Given two surfaces with similar topology, it is possible to compute a one-to-one correspondence between them [13]. The problem of computing such a mapping is referred to as surface parameterization.

Conformal mapping is one of the surface parameterization techniques that preserves both angles and shapes.Let a continuous surface $S$ in 3D space be parameterized into a parametric domain $\Omega \subset \mathbb{R}^2$. As illustrated in Fig. 3, a function $f$ mapping from the $\Omega$ domain in 2D to a 3D surface $S$ is said to be conformal if for each $(u, v) \in \Omega$, the tangent vectors along the horizontal $\nabla u$ and vertical lines $\nabla v$ (the red and blue lines in Fig. 3), forming a regular grid, are orthogonal on $S$ and have the same norm [14]:

$$\nabla v = \mathbf{n} \times \nabla u, \tag{3}$$

where $\mathbf{n}$ denotes the unit normal to the surface $S$. In other words, a conformal mapping locally corresponds to a similarity transform - *i.e.,* transforms an elementary circle of the $(u, v)$ domain to an elementary circle on the surface.

### B. Surface Drawing with Minimal Distortion

To realize conformal mapping in our work, we adopted the least squares conformal mapping (LSCM) [16] to parameterize the target surface, that is based on energy minimization on the non-conformality of the mapping function. Once we *unfold* the target surface into 2D parameter space $(u, v) \in \Omega$, we search for proper parameter values of the 2D drawing data in the parameter space, and *refold* the surface into 3D space. As mentioned in Sec. IV-B, surface normals could be estimated in different ways. Accordingly, conformal mapping could be done differently as well. We choose to compute gradients with a local orthonormal basis of triangles, which needs the surface to be reconstructed as a triangular mesh. Alternatively, one could use a meshless technique for conformal mapping of a point cloud without surface reconstruction using Laplace-Beltrami (LB) operator [24]. Both results would yield a set of coordinates $(u_i, v_i) \in \Omega$ associated with each point in $P$ that satisfies Eq. 3. In our implementation, we have chosen the meshing technique, as robust surface meshing implementations are readily available for use [25] and it is easier to implement this way.

After the surface parameterization is done, we need to decide the proper parameter coordinates for the control points set $C$, which will then be mapped back to 3D using the parameterization. Since our captured point cloud set $P$ is an unorganized point set and does not necessarily form a uniform grid, in order to parameterize every control point to a corresponding $(u, v)$ coordinate, we need to solve an inverse mapping/parameterization problem. Note that this is not a big issue for raster-based texture mapping, as one can have a clear idea of which rasterized point on the surface needs to be parameterized. On the other hand, in our case, the control points are originally defined in 2D, and we have no idea where these points will be mapped to the 3D surface. To solve this inverse mapping problem, we simply perform bi-linear interpolation in the parametric domain to estimate $(u, v)$'s for $C$ as follows.

Given a desired drawing space $\mathcal{W}$ in 3D along with the desired drawing scale, we can compute the parametric scale in $\Omega$ and fit the 2D drawing to $\Omega$. Then, for each control point $\mathbf{c}_i \in C$, we search for the four-nearest points in $P$ that form a quadrilateral in $\Omega$ that contains $\mathbf{c}_i$. By performing bi-linear interpolation on this quadrilateral using our previous work [6], we can parameterize $\mathbf{c}_i$ that is mapped to both the position $\mathbf{p}_i$ and the surface normal $\mathbf{n}_i$ on $P$. Note that we calculate the surface orientation $\mathbf{r}_i$ only for the control points of drawing, not for all the points on the surface, which is more dense than the set of control points. Finally, a set of control points mapped to the target surface with the corresponding surface orientations, $C' = \{\mathbf{c}'_i = (\mathbf{p}_i, \mathbf{r}_i) | \mathbf{p}_i \in \mathbb{R}^3, \mathbf{r}_i \in SO(3)\}$, is generated.

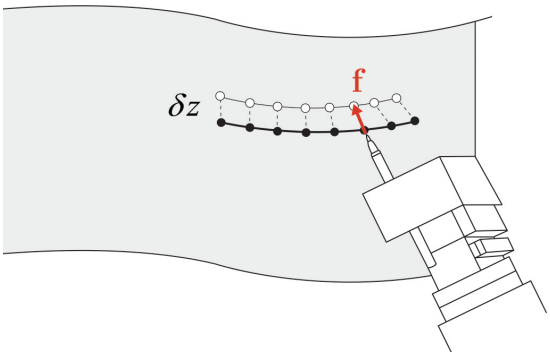
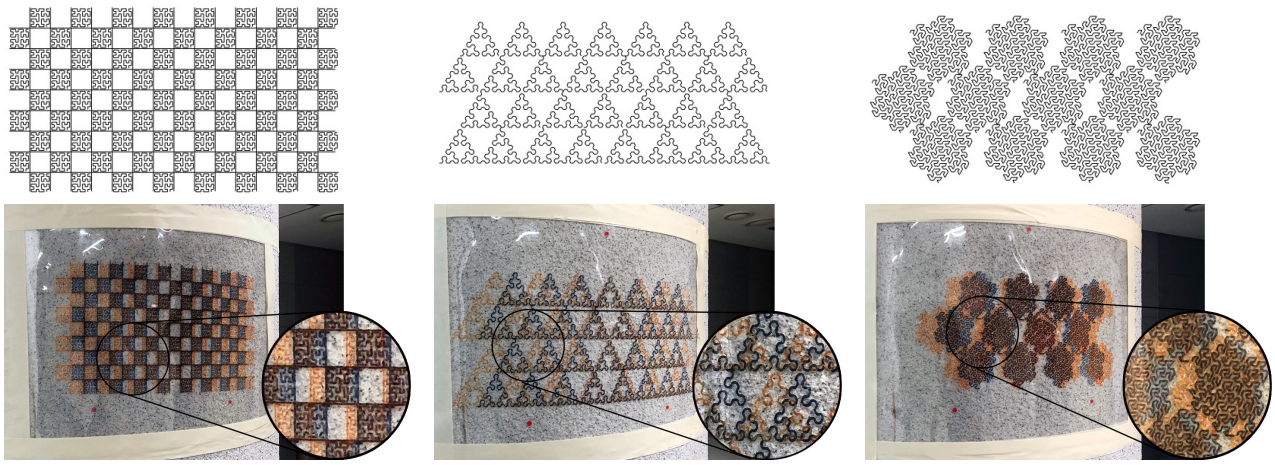



Fig. 4. Compliant force ($\mathbf{f}$) is generated by the impedance-controlled manipulator. The thick black line represents spline curves on the physical surface that need to be drawn with a set of black dots for control points, also corresponding to the origin of the pen frame $\mathcal{P}$. Meanwhile, the thin line with a set of circular dots represents the target spline curves on the virtual surface, located slightly under the physical surface by $\delta z$, corresponding to the origin of the virtual frame $\mathcal{V}$. The orientations of $\mathcal{P}$ and $\mathcal{V}$ are identical but their z-axes are offset by $\delta z$.



(a) Grid with Hilbert space-filling curve (b) Sierpiński arrowhead curve (c) Koch snowflake curve

Fig. 5. Drawing Results on a Bumpy, Circular Column Wall. The first row shows the original fractal curves in 2D including Hilbert space-filling curve, Sierpiński arrowhead curve, and Koch snowflake curve from left to right. The second row shows the robotic drawing results, the black lines using our method, and orange lines using projection mapping. Note that the orange lines are distorted compared to the original2D drawings.

## VI. IMPEDANCE-CONTROLLED DRAWING

Using surface estimation and mapping, our drawing robot is now fully provided with a set of Bézier curves in 3D that can be drawn on the target surface. However, to exert a proper compliant force at a pen-tip (the end-effector) as well as to compensate for possible depth-estimation and calibration errors, we adopted an impedance control method. Impedance control is one of the hybrid, position- and force- controlled method that was proposed to interact with an unstructured environment [18]. By employing a mass-spring-damper-like system, the impedance control allows a robot to react in a compliant manner to external obstacles. By considering a certain offset value (*i.e.,* impedance) for each control point, corresponding to the estimation errors, the impedance control results in continuous contact motions with the surface during the entire drawing session.

We configured the impedance controller in such a way that the robot is compliant only in the normal direction of the surface, as the pen tip attached to the robot manipulator is oriented oppositely toward the estimated surface normal. Additionally, in order to exert an appropriate amount of compliant force at a pen-tip, a small deviation between the target position and the physical position of the pen-tip needs to be provided to the impedance controller.

Simply having the control points set $C' = \{\mathbf{c'}_i = (\mathbf{p}_i, \mathbf{r}_i)\}$ define the target drawing frames can result in lack of sufficient pen pressure and will be very sensitive to the surface estimation error. Therefore, the target position $\mathbf{p}_i$ of $\mathbf{c'}_i$ is modified to:

$$\mathbf{p}'_i = \mathbf{p}_i + \delta z \mathbf{n}_i, \tag{4}$$

where $\delta z$ is a user-defined gain value that controls the pen pressure and $\mathbf{n}_i$ is parallel to the z-axis $\mathcal{P}_z$ of the pen frame $\mathcal{P}$. This deviation results in a compliant force $\mathbf{f}_z = k\delta z$ along $\mathcal{P}_z$, where $k$ is the spring stiffness. To be more efficient, instead of calculating new positions for every $\mathbf{c'}_i$, we attach a virtual frame $\mathcal{V}$, as shown in Fig. 1-(c), to the physical pen aligned with the pen-tip frame $\mathcal{P}$ attached to the end of the pen except that $\mathcal{V}$ is slightly offset by $\delta z$ from $\mathcal{P}$ along z-direction, which has the same effect of moving $\mathcal{P}$ to $\mathbf{p}'_i$.

As a result, the pen-type end-effector traces out the position of spline curves while maintaining the contact with the surface, exerting an almost uniform amount of compliant forces regardless of the shape of the surface.

## VII. RESULTS AND DISCUSSIONS

### A. Implementation Details

As shown in Fig. 1-(a), our robotic surface drawing system consists of a KUKA LBR IIWA 7 R800 manipulator equipped with a 3D-printed gripper as an end-effector to hold various types of pens that is solid enough to exert a force. We use an Intel RealSense ZR300 RGB-D camera to capture 3D point cloud of the target surface. It is a stereotype camera that is augmented with an infrared projection system to detect mono-colored objects more precisely.

We use C++, Java and MATLAB with a PC equipped with 6 cores Intel Xeon E5 CPU and a 16-GB RAM under Windows 10 64bit and Ubuntu 16.04 64-bit operating systems. We also use Point Cloud Library (PCL) [25] to generate point cloud data and to reconstruct a triangular mesh from the point cloud.

To benchmark our robotic drawing system, we used two types of input drawings: patterned drawings and artistic drawings to effectively show both distortion-free and complicated pen drawing results. For patterned drawings, we generated several fractal curves including a Hilbert space-filling curve, a Sierpiński arrowhead curve, and Koch snowflake curves that form uniform squares, triangles, and hexagonal shapes. Artistic drawing data sets in Fig. 6 are acquired by using the vector graphics engine proposed in our previous system [6].

### B. Experimental Results

Fig. 5 shows examples of the robotic pen drawing results using our system on a bumpy circular column wall with patterned data sets in black compared to a simple projection mapping used in [6] in orange. Compared to the results using projection mapping, our method reproduces the original drawing faithfully on the surface. On the other hand, the projection mapping method does not preserve the original length and it gradually increases as it moves away from the center of projection. As can be seen from the grid pattern results shown in Fig. 5-(a), our method preserves the side lengths of the grid, but in projection mapping, the length has increased by more than 20% in the worst case.

The point cloud of the target surface is generated in $480 \times 360$ resolution, generating over 172K points for a typical scene. The statistics of our experimental results including the number of control points, the size of the mapping surface (*i.e.,* canvas size), the mapping calculation time and the robot drawing-execution time are provided in Table I. The drawings can be stretched into any smaller, or bigger sizes only if it is within the robot's workspace. Even though we parallelize some of the mapping tasks, the drawing time can be further reduced by adopting a few acceleration techniques - for instance, a more efficient neighborhood search method with an efficient data structure such as Delaunay triangulation.

TABLE I
EXPERIMENTAL STATISTICS

| Drawings | (a) | (b) | (c) |
|---|---|---|---|
| # of Control points | 6,930 | 2,360 | 4,128 |
| Mapping Surface Size (mm) | 384×216 | 432×216 | 384×192 |
| Mapping Time (sec.) | 635 | 308 | 523 |
| Execution Time (min.) | 54 | 14 | 19 |

We also show experimental results of artistic drawings in Fig. 6, drawn using vector graphics engine. The drawings contain over 90K control points, and the robotic drawing on a circular column wall took about 5 hours for each drawing.

### C. Discussion

Maintaining continuous contact with an arbitrary surface with an intended amount of contact force is not an easy robotic task. Our system, however, is able to perform robotic surface drawing with nearly constant compliant forces, while minimizing the distortion of the original input vector drawings.

Currently, we rely on the depth information captured using an RGB-D camera so that our system is not suitable for a transparent or a reflective surface, such as on a window and on a mirror. Additionally, the depth and normal estimation process can be greatly accelerated by adopting a more efficient nearest neighborhood searching method, such as ANN [26]. Another problem is that our robot has a limited workspace, even though the input drawing consisting of a sequence of vectors could be scaled indefinitely. We want to address this issue in the future by adding mobility to our current manipulator.

Fig. 6. Artistic Drawing Results

## VIII. CONCLUSION

We presented a robotic surface drawing system that can generate pen drawings on physical surfaces with minimal drawing distortion, realized by conformal mapping and effective surface parameterization. Our distortion-free mapping method successfully maps a sequence of 2D vectors to 3D real-world space. With impedance control, we compensate for possible estimation or calibration error and generate continuous contact motions. The experimental results show that our system has the potential to be extended into other robotic applications than drawing that requires a robot to follow a given trajectory while maintaining contact with the underlying surface.


## ACKNOWLEDGEMENTS

This project was supported by the National Research Foundation(NRF) in South Korea (2017R1A2B3012701).


## REFERENCES


[1] J. Reichardt, "Machines and art," *Leonardo*, vol. 20, no. 4, pp. 367–372, 1987.

[2] S. Calinon, J. Epiney, and A. Billard, "A humanoid robot drawing human portraits," in *IEEE-RAS International Conference on Humanoid Robots*, 2005.

[3] T. Lindemeier, S. Pirk, and O. Deussen, "Image stylization with a painting machine using semantic hints," *Computers and Graphics*, vol. 37, 2013.

[4] P. Tresset and F. F. Leymarie, "Portrait drawing by paul the robot," *Computers and Graphics*, vol. 37, 2013.

[5] S. Damith Herath, Christian Kroos, *Robots and Art*. Springer, 2016.

[6] D. Song, T. Lee, and Y. J. Kim, "Artistic pen drawing on an arbitrary surface using an impedance-controlled robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[7] C. Hofner and G. Schmidt, "Path planning and guidance techniques for an autonomous mobile cleaning robot," *Robotics and Autonomous Systems*, vol. 14, no. 2, pp. 199 – 212, 1995, research on Autonomous Mobile Robots. [Online]. Available: http://www.sciencedirect.com/science/article/pii/092188909400034Y

[8] Y.-B. Jia, L. Mi, and J. Tian, "Surface patch reconstruction via curve sampling," in *IEEE International Conference on Robotics and Automation*, May 2006, pp. 1371–1377.

[9] J.-D. Boissonnat, L. J. Guibas, and S. Oudot, "Learning smooth objects by probing," in *Proceedings of the Twenty-first Annual Symposium on Computational Geometry*, ser. SCG '05. New York, NY, USA: ACM, 2005, pp. 198–207. [Online]. Available: http://doi.acm.org/10.1145/1064092.1064124

[10] P. McCorduck, *AARON'S CODE: Meta-Art, Artificial Intelligence, and the Work of Harold Cohen*. W. H. Freeman & Co, 1990.

[11] M. Stein and C. Madden, "The pumapaint project: Long term usage trends and the move to three dimensions," in *Proceedings of the IEEE Intl Conference on Robotics and Automation (ICRA)*, 2005.

[12] B. Galea, E. Kia, N. Aird, and P. G. Kry, "Stippling with aerial robots," in *Computational Aesthetics in Graphics, Visualization and Imaging*, 2016.

[13] K. Hormann, B. Lévy, and A. Sheffer, "Mesh parameterization: Theory and practice," 2007.

[14] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon mesh processing*. AK Peters/CRC Press, 2010.

[15] W. T. Tutte, "Convex representation of graphs," in *London Mathematical Society*, 1960.

[16] B. Lévy, S. Petitjean, N. Ray, and J. Maillot, "Least squares conformal maps for automatic texture atlas generation," in *ACM transactions on graphics (TOG)*, vol. 21, no. 3, 2002, pp. 362–371.

[17] A. Sheffer and E. de Sturler, "Parameterization of faceted surfaces for meshing using angle-based flattening," *Engineering with Computers*, vol. 17, no. 3, pp. 326–337, Oct 2001. [Online]. Available: https://doi.org/10.1007/PL00013391

[18] N. Hogan, "Impedance control: An approach to manipulation," in *American Control Conference, 1984*. IEEE, 1984, pp. 304–313.

[19] S. A. Schneider and R. H. Cannon, "Object impedance control for cooperative manipulation: Theory and experimental results," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 383–394, 1992.

[20] F. Caccavale, C. Natale, B. Siciliano, and L. Villani, "Six-dof impedance control based on angle/axis representations," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 289–300, 1999.

[21] T. Wimbock, C. Ott, and G. Hirzinger, "Impedance behaviors for two-handed manipulation: Design and experiments," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, April 2007, pp. 4182–4189.

[22] J. Lee, P. H. Chang, and R. S. Jamisola, "Relative impedance control for dual-arm robots performing asymmetric bimanual tasks," *IEEE transactions on industrial electronics*, vol. 61, no. 7, pp. 3786–3796, 2014.

[23] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," *KI-Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, 2010.

[24] J. Liang, R. Lai, T. W. Wong, and H. Zhao, "Geometric understanding of point clouds using laplace-beltrami operator," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 214–221.

[25] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–4.

[26] D. M. Mount and S. Arya, "ANN: library for approximate nearest neighbour searching," 1998.