Towards 3D Perception and Closed-Loop Control for 3D Construction Printing

Xuchu Xu, Ruoyu Wang, Qiming Cao, Chen Feng

Tandon School of Engineering, New York University, Brooklyn, NY 11201, USA

{xuchu.xu, ruoyuwang, qimingcao, cfeng}@nyu.edu

Abstract -

With the expanding size of additive manufacturing products, research pioneers start to explore 3D printing in the construction field. For 3D construction printing, quality means safety, cost, and efficiency. However, it is challenging to ensure quality via defect detection and deviation correction during the construction printing process. Conventionally, defect and deviation still rely on the quality check after printing is completed or on-site manual monitoring, which could cause either a waste of material and time to abort printing or lagging adjustment for printing settings after obvious defect appeared. To overcome these challenges, we propose a point-cloud-based approach for real-time 3D construction printing defect detection using a 3D camera and cloud-to-plane distance to evaluate printing layer integrity and compare printing results with CAD models. We also define different types of defects and deviations that can cause printing failure. Additionally, we feedback detection output into a closed-loop controller for updating the printhead motion. Our experiments show this joint printing and detection process handling various defects and deviations.

Keywords -

3D Construction Printing; Point Cloud Comparison; Defect Detection; Close-loop Control

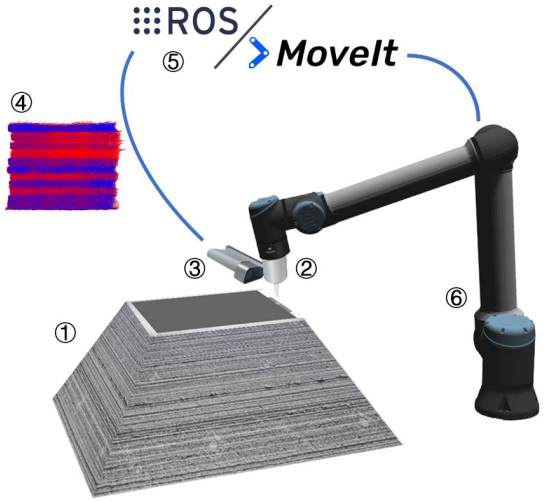

Figure 1. Illustration of our setup. 1. Concrete printing model. 2. Print head. 3. 3D perception sensor. 4. Color-coded printing error. 5. Control system. 6. Robot arm. During the printing process, the 3D perception sensor updates error feedback to the control system for defect and deviation correction.

1 Introduction

During the last several decades, additive manufacturing (AM), also widely known as 3D printing, demonstrated an incredible ability to assist designers and engineers prototype and produce rapidly. Recently, the study of 3D construction printing has rapidly risen as a new active area in the AM communities. A key challenge in 3D construction printing is monitoring the output quality and in real-time automatically adjusting the printhead control to reduce deviation and avoid structure collapse. To achieve this, 3D perception should to be tightly coupled into the printing process, since we need to evaluate not only each layer's integrity but also the evolving structure's shape deviation from the CAD model.

To acquire 3D shape data, traditional sensors, like multi-beam Lidar [1, 2], firmly occupy the high-end manufacturing and research of 3D perception. Although it could provide highly accurate point cloud data, a notable drawback of this type sensor is the excessive cost. Another limitation of this sensor in 3D printing is the difficulty of point cloud segmentation in post-processing due to their spare coverage on the printing output. To overcome these drawbacks, a different approach is to combine Time-Of-Flight (TOF) sensor with RGB camera. Microsoft Azure Kinect (Kinect) benefit from its high resolution TOF sensor, which has a balanced cost and performance. Kinect could provide precise and densely distributed point cloud data with RGB information that help us extract useful points while easily excluding the insignificant background. Moreover, we can observe the printer's extruder from Kinect's RGB camera in real time, which monitors material without interrupting the printing process.

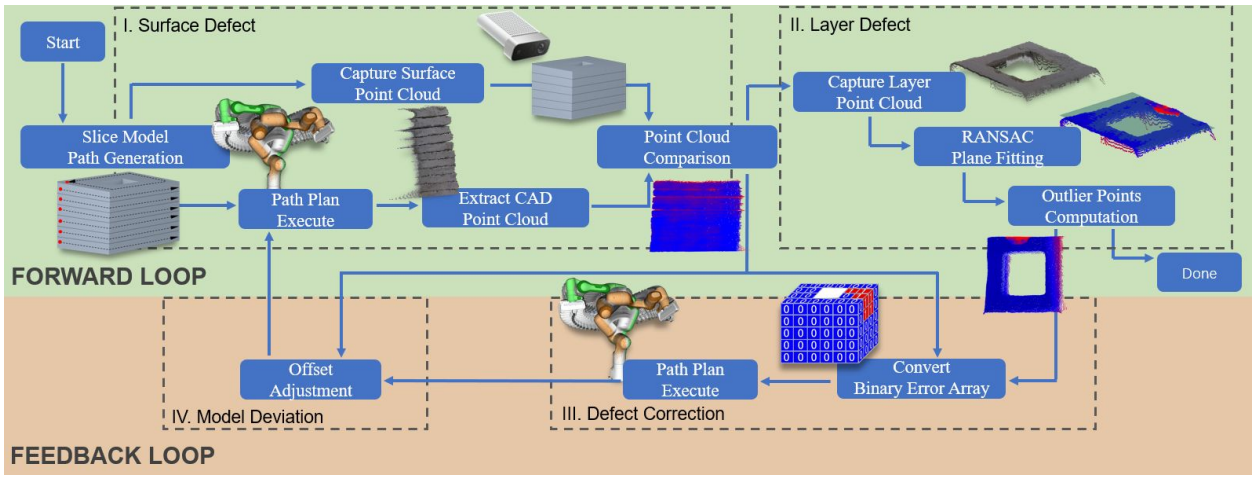

Figure 2. Our workflow. I. Simultaneous surface defect detection and 3D printing. II. Defect detection after printing the current layer. III. Error feedback. IV. Error adjustment from the current to the next layer.

With dense RGB point cloud data, we will measure surface defect and shape deviation compared with the corresponding point cloud which is generated from the CAD model. Cloud-to-plane (C2P) [3] distance is the error metric we adopt for measuring the difference between two point clouds. By projecting error vectors along the unit normal vector of the local plane, this metric could evaluate shape deviation and trigger with preset tolerances the online printing correction through feedback control.

In this paper, we propose an online approach for increasing the quality of 3D construction printing using a 3D camera and feedback control. To this end, we designed two experiments testing several different types of defects and deviation. Due to COVID-19 pandemic and facility closure, one of our experiment, closed-loop 3D printing, was only conducted in the simulation environment. The following are our main contributions in this paper:

- We define three types of printing error in 3D construction printing, includes layer defect and deviation, surface defect, and model deviation.
- We propose a novel approach for closed-loop 3D construction printing that could help us correct defects during the printing process.
- We implement a cloud-to-plane shape deviation error assessment between the point cloud captured by the 3D camera and that generated from the CAD model.
- We design two experiments to demonstrate our approach under different printing error scenarios.

2 Related Works

Our approach is mainly involved in three major research domains: 3D construction printing, defect detection and closed-loop control printing. As our previously mentioned, point cloud matching algorithm employed to find corresponding points between two point clouds and help us to calculate the error rate between them. Therefore, we will expand to discuss related research and approaches in these three areas.

3D Construction Printing. Different from other 3D printing types, construction 3D printing needs more consideration in product transportation, material property, printing quality and even aesthetic requirement. Crump et al. [4], as inventor of fused deposition modeling (FDM) technology, opened the gate of 3D printing. His method slices 3D objects to 2D layers and prints by CNC based machine. Although his approach initially used polymer filament, this method is also applicable to use other materials in different regions, such as AM with concrete material. Contour crafting [5, 6], present by Khoshnevis, is one of approach which employed for 3D construction printing. This approach sets a side trowel on the side of the extruder to smooth the apparent finish texture produced by the printing process. Buswell et al. [7] proposed the concept of Freeform Construction, which brings 3D printing into mega-scale rapid manufacturing. He aims to divide the building into a mega-scale section that could rapidly produce in the factory and assemble at the construction site. Recently, Keating et al. [8] demonstrated a mobile printing platform called Digital Construction Platform (DCP). DCP used a KUKA robot arm to research the printing position instead of the gantry-base fixed platform. Also, the author of DCP uses two-component polyurethane closed-cell foam to ensure printing material curing in a short period of time. 3D construction printing is not only a popular topic in the research area but also favored by the construction industry. WinSun [9], a Chinese 3D print-

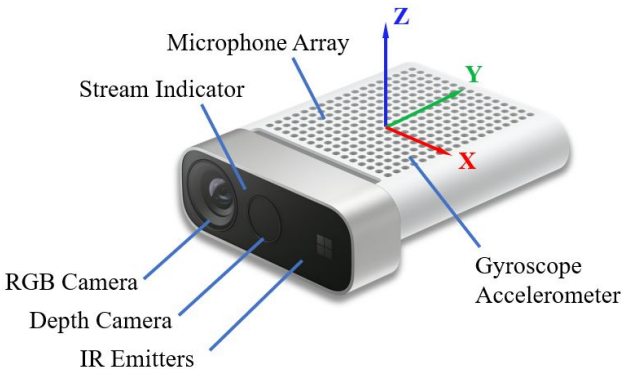

Figure 3. Microsoft Azure Kinect. We use RGB camera, Depth camera and Gyroscope in the following experiments.

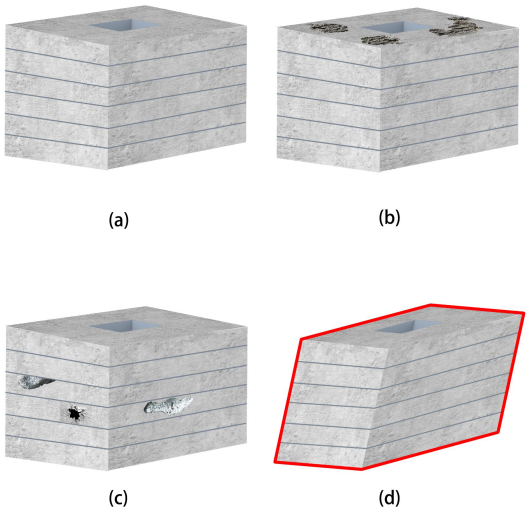

Figure 4. Defect and Deviation Diagram: (a) Ground Truth; (b) Layer Defect and Deviation; (c) Surface Defect; (d) Model Deviation.

ing industry leader company, developed multiply property material and successfully applied 3D printing to the construction field. Therefore, we need to explore a reliable method to real-time monitor the quality of 3D printing.

Defect Detection. Currently, defect detection methods of 3D Printing fall in two different domains, scanning based and vision based. Scanning based methods usually use a TOF sensor or with additional auxiliary equipment. Benefit form scanning whole model, these type methods could inspect detailing defects, such as tiny surface crack and local deformation. One of the obvious drawbacks is time-consuming, that these methods required a screening window to scan each layer during the printing process. Moreover, the calibration and preciseness of auxiliary equipment will directly affect the final accuracy. Lin et al. [10] used a sliding window to detect filling situations by a laser scanner, which attaching with printer extruder. Liu et al. [11] used a camera to measure the target surface assisting with a line laser and linear translation stage. Image based methods only rely on different types of cameras. Holzmond et al. [12] demonstrated real time defect monitor system that compares layer point cloud with a model cross section by using a dual camera under different light sources. Shen et al. [13] proposed a feature based surface defect detection approach by contour comparison. There are also some vision based methods combined with neural network [14, 15]. The downside is that these methods require a large amount of data, especially labeled data for supervised learning. Moreover, most of the methods above are implemented in polymer FDM 3D printing. Either point cloud of one single layer is too sparse or model defects is difficult to be accurately captured online. Therefore, all of the above methods only detect defects after printing and can not correct error during printing process.

Closed-loop Control Printing. Nowadays, researchers have not only focused on developing new 3D printing technology, but also hoped to improve the quality and automation level of 3D printing, which reducing the degree and impact of human factors. Different types of 3D printing have disparate limitations and drawbacks which need to overcome. For the ink-jet printer, Lu et al. [16] proposed a feedback controller for finding the best droplet location to minimize the edge shrinkage effect. Guo et al. [17] incorporated feedback measurements to a predictive control algorithm for avoiding edge shrinking, unreliable dimensions and uneven surfaces. Altın et al. [18] developed a spatial iterative learning control framework that involves discrete Fourier transforms and iterative learning control to improve part's build quality from a single layer toward the relationship between two layers. For laser metal deposition (LMD) printing, Sammons and his colleagues [19] present a stabilizing layer-to-layer controller to track and compensate for the deposition process. When we ask why closed-loop control is so important in 3D printing, especially used in manufacturing, we need to understand that the nature of 3D printing is repetitive motion. The quality of the current layer is highly dependent on the previous layer, such as integrity, flatness and support structure.

3 Method

As shown in Figure 2, the entire 3D printing processing workflow is divided into four phases. Phase 1 is mainly composed with the regular 3D printing process and surface defect detection, which also includes layer deviation detection. Phase 2 includes our original steps, which are layer defect detection after each layer printing. Phase 3 aims to integrate the error output of phase 1 and 2, then following to repair the model. The goal of phase 4 is to

compensate for model shape deviation by adjusting offset. The detail of the explanation and discussion will present in section 3.2.

3.1 Hardware Platform

Perception Sensor. Microsoft Azure Kinect, Figure 3, is a fusion sensor that provides an RGB camera, infrared sensor, depth sensor, gyroscope with accelerometer and microphone array. The reason why we chose Kinect as perception sensor in our approach is that it preset synchronization and calibration-free transformation between each sensor. The depth sensor model in our research is set as WFOV 2x2 binned. The nominal range accuracy of Kinect is less than 11mm in the distance range of 0.25-2.88m [20]. It provides 512x512 resolution with 0.5m to 5m operating range and 120 degrees field of interest in the dual-axis, accurate and dense enough for our purpose.

3D Printing Platform. Universal Robot 10e (UR10e) [21] is a 6-DOF collaborative robot arm that could handle up to 10 kg payload in a 1.3 meters radius and 360 degrees workspace. We designed a concrete print head with a center mixer and attachment for Kinect. The center mixer could rotate clockwise to feed material and the opposite direction to block material. We prepared fast setting mortar mix as our printing material.

Control and Simulation Environment. In order to better observe printing process, ensure safety environment and prevent hardware damage, we use ROS [22] with Moveit [23] in the Gazebo environment to simulate the printing path. Moreover, we use Gazebo simulation world and physics engine for simulating and demonstrating the closed-loop control experiment in section 4.3.

3.2 Defect and Deviation Detection.

When the failure of the 3D printing model occurs, defect and deviation will appear in some places of our model. Therefore, we need to clarify what is defect and what is deviation. In this paper, we define a defect as missing the integrity of the printing layer or material overfill and underfill on the printing model surface. Similarly, the deviation is defined as the contour or shape error.

We divide the defect and deviation generated in 3D construction printing into the following three categories:

Layer Defect and Deviation. Since the printing model is stacked up by multiply concrete layers, it is necessary to check the defect and contour deviation of each layer. The cause of layer defect is either the interruption of feeding material or the impurities contained in the material, such as air bubbles. Layer deviation is caused by path planning error or robot arm control accuracy and noise.

We use RANSAC to fit a plane, $\hat{P}_L$, based on the points in the point cloud, S_L, captured from the Kinect sensor.

$$\begin{aligned} S_L &= \{(x, y, z)\} \\ \hat{P}_L &= \{(x, y, z) | \hat{A}x + \hat{B}y + \hat{C}z + \hat{D} = 0\} \end{aligned} \tag{1}$$

By calculating the point to plane distance, d_i, of each points in the point cloud, we uses an error index array, $e(d_i)$, to record those outliers which compare with the preset tolerance, λ.

$$d_i = \frac{|\hat{A}x_i + \hat{B}y_i + \hat{C}z_i + \hat{D}|}{\sqrt{(A^2 + B^2 + C^2)}} \tag{2}$$

$$e_L(d_i) = \begin{cases} 1 & |d_i| \geq \lambda \\ 0 & |d_i| < \lambda \end{cases} \tag{3}$$

Surface Defect. The surface of the 3D printing model is formed by stacking the sides of the layer. Hence, the obvious layer texture can be observed on the model surface. During the printing process, due to changes in the print head's motion speed or angle rotation, it will cause the material to accumulate at some particular positions, thus forming a class of overfill defect. Unlike overfill, underfill is happened at the gap between roaster path and contour path. Additionally, the local collapse by lacking support will also cause surface underfill.

In order to slice the CAD model for outputting each layer's outer surface, we export the CAD design file to STL format. During printing operation, we subscribe ROS message which present Kinect's position in global coordinate system by calculating robot's forward kinematics. After that, we use this position to map points cloud into Kinect coordinate system. By applying Iterative Closest Point (ICP) algorithm, we identify corresponding point, p_i, for each point, $\hat{p}_i$, between CAD point cloud and scanning point cloud respectively. We minus p_i and $\hat{p}_i$ to get error vector. Then we project the error vector along the unit norm vector N_p on point p_i in reference point cloud P. Therefore, the C2P error distance, d_i, is finally computed as,

$$d_i = \min_{\forall p_i \in P_s} (|\hat{p}_i - p_i| \cdot N_p) \tag{4}$$

Once we have the C2P error distance, we apply (3) to compute point cloud error array.

Model Deviation. Even our printing model avoiding previous errors, we may still fall short of our printing results. The shape deviation we discuss here refers to the misalignment between layers. To provide misalignment bias to feedback control system, here we choose absolute error distance instead of error index array in (3).

$$e_M(d_i) = \begin{cases} d_i & |d_i| \geq \lambda \\ 0 & |d_i| < \lambda \end{cases} \tag{5}$$

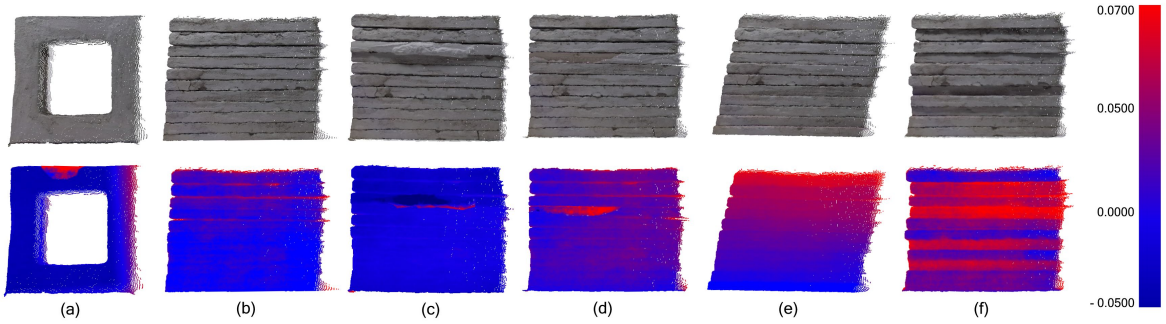

Figure 5. Task 1 results: (a) Layer Defect; (b) Surface without Defect and Deviation; (c) & (d) Surface Defect; (e) & (f) Model Deviation. Top: RGB point cloud captured by Kinect. Bottom: visualization of C2P error between scanned point cloud and CAD. Signed error is color coded from red (positive) to blue (negative).

3.3 Closed Loop Control

As we mentioned in the beginning of this section, we explain and discuss in detail for the each phase setup and process below:

Phase 1. First of all, we directly export the design model from CAD drawing software and save it in STL format. Then we send this STL file to ROS Additive Manufacturing (RAM) for 3D slicing and path generation. We import each layer path from RAM output into ROS MoveIt! to perform path planning. This step helps us to ensure UR10e avoiding self-collision. Once MoveIt! does not detect any collision, it will send the trajectory to UR10e and execute.

At the same time, our point cloud comparison algorithm is also working as a ROS node during the printing process. By subscribing pose message, ROS transfer the position of UR10e's end effector to point cloud generation. In our algorithm, we compute the Kinect observation frame and generate the reference point cloud under the current frame. On the other side, the Kinect sensor continuously captures depth images of the layer surface and transmits it back to our algorithm in real-time. When UR10e completes printing each layer, our approach will go through the ICP algorithm to register the scanning point cloud to reference point cloud and calculate the C2P distance to output the error-index array.

Phase 2. During exporting error-index array in phase 1, UR10e move forward to the next phase, which set Kinect to capture the top view of each layer. By converting RGB to HSV color space, we able to easily segment layer's points from the scanning point cloud. Subsequently, we apply the RANSAC algorithm to fit a plane from those layer's points and output as the planar equation format. The manipulation of the planar equation can help us check multiple layer defects. We check the layer's level corresponding to the horizontal datum. We also find the outlier compare with our preset distance to find the defect position on the layer plane. The same as phase 1's last step, the algorithm outputs the error-index array to provide feedback information.

Figure 6. Task 1 results. Left: RGB point cloud captured by Kinect. Right: visualization of Layer Deviation between scanning point cloud and CAD file. Signed error is color coded from red (positive) to blue (negative).

Phase 3. In the previous two phases, we got error-index arrays. In the feedback loop, we first convert them into binary error-index arrays to indicate the makeup position for MoveIt! path planning. During MoveIt! execute the trajectory path, we able to correct layer deviation before concrete curing by setting the different angles of side trowel. Moreover, for those defect positions, the print head will re-feed concrete with adjusted side trowel.

Phase 4. Benefit from error array provided in phase 1, the final phase is able to estimate the offset between the current layer and previous layers on each axis direction. In this step, we regard the previous layer as an integral base and only discuss the relationship between the current layer and the base. Applying compensation could help us avoid systematic bias, even compensates control noise for certain special position.

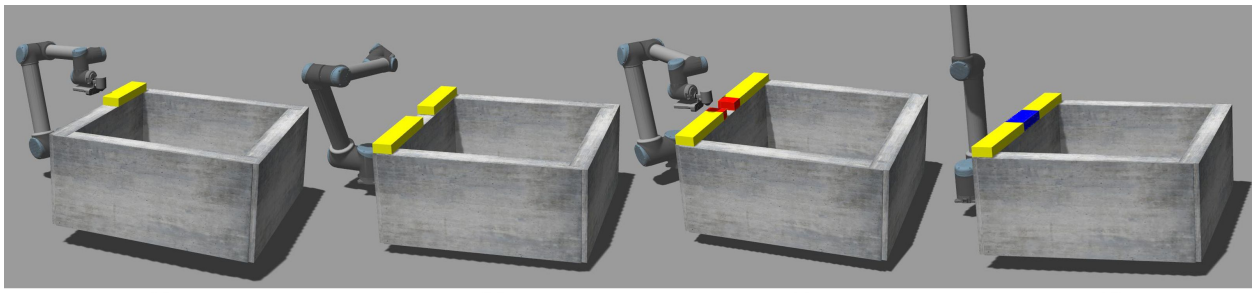

Figure 7. Task 2 experiment process: (a) Normal Printing without Defects; (b) Printing Complete; (c) Mark Defect Area; (d) Complete Defect Correction.

4 Experiments

Our experiment is designed as two tasks, static detection test, and simulation of 3D construction printing. The first task, static detection, aims to demonstrate our approach worked as normal defect detection in a static environment. We will show the three different defects and deviations, which defined in section 3.2. For the second task, we will present our complete approach including defect detection with feedback control during real-time printing. Due to research facility closure and limited available equipment, this experiment will be present in the simulation world by Gazebo.

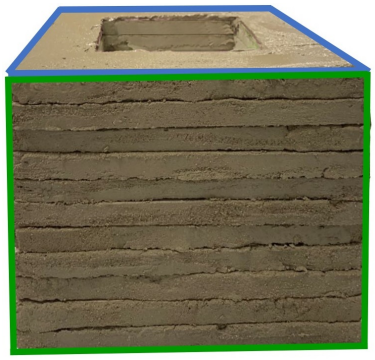

Figure 8. Eleven-layer concrete cubic. Top (Blue): Layer without defect and deviation. Front (Green): Surface without defects.

4.1 Experiments Setup

In Task 1, we are looking for a model that would demonstrate errors as mentioned in section 3.2. For better simulate the product of 3D construction printing, including material property and texture, we made serval half inches concrete planes by concrete-water mixture in a ratio of 9:1. We superimposed these concrete planes to build a cubic shape model. Here, each concrete plane represents corresponding layer in the real printing process. Therefore, we finally got a cube that is 6.5 inches high and 8.5 inches in both depth and width. Each concrete plane has two sides of fixed length and 90 degrees angle, the same as our CAD design. The other two sides are designed to contain different deviation and defect, such as material overfill or layer deviation. We can simulate the errors under various situations by adjusting the placement and order of concrete planes, just like we designed in advance.

In Task2, we setup a UR10e robot with a combination of a simple print head an kinect in Gazebo's simulation environment. To prevent the robot arm hitting the ground, we set the UR10e on a 0.7m high box. Similarly, the printing platform also raised 0.4m accordingly. In this task, our goal is to use the Kinect to capture the layer defect caused by interrupt feeding material during the printing process. Since there is no fluid setting in Gazebo, we made a plugin that will follow the print head position to place thin yellow boxes from the waiting area to the target location in real time. We randomly deleted one segment of materials in the waiting area. Therefore, UR10e will interrupt printing even robot arm is still moving. By using the depth camera plugin, we convert the depth image to a point cloud and use it as the feedback input of our approach. After the printing complete, our plugin will mark the defect position in red first and turn blue after defect correction.

4.2 Defect and Deviation Detection

Figure 5 and Figure 6 present the experiment results in task 1. For Figure 5, RGB point cloud images in the top row demonstrate the concrete cubic that we see in the real world. The images in the bottom row show the difference between printing model and CAD design by presenting in heat map. As scale bar shows, the red regions represent the shrinkage of the model, which means that these regions do not meet the designed size. Similarly, the dark blue areas tell us that the printing exceeds CAD design. As mentioned in section 3.2, tolerance is always a necessary consideration in the engineering world. We set 0.5cm as

the tolerance, λ, in (3) and (5).

The images in column (a) are captured by Kinect from top to bottom after completing each layer. We can see that there are only slight traces in the top image, even it will not be noticed, if the observer does not look closely. However, after generating RGB point cloud, we can easily segment our target, concrete cubic, from the background by color space change. The heatmap below makes us more intuitive to see the defects that are difficult to detect with naked eye or image based detection approaches. There is a shrinkage area at the top of the model. In the actual printing process, the reason of causing this type failure could be the interruption of feeding material. Furthermore, we can also determine the layer deviation at the same time. In Figure 6, the red area presents layer deviation exceeding the boundary of the CAD design model.

Column (b) in Figure 5 shows an ideal 3D printing model that we can see that the entire point cloud is the blue color. Columns (c) and (d) simulate two types of surface defects, overfill and underfill, respectively. In the top image of column (c), we can see a prominent material overflow part in the middle of our printing model. When we applied our method, the dark blue zone is the detection of the overfill area. The note that the underfill depth in column (d) is only 0.7 cm, but our approach still can accurately detect it. For model deviation detection, we simulate two types of control error, columns (e) and (f), that cause to misalignment between neighbor layers. Column (e) simulates the printing drift, which usually occurs in the control systems with accumulated errors. We choose random noise for offset error and make the detection in column (f).

4.3 Closed-loop Control Printing

As Figure 8 shown, our goal is to print a 1 meter cubic. The initial experiment randomly deletes material for an arbitrary length. After confirming that our experiment plan is feasible in the simulation environment, we change the range of the interrupted material to check the limitation of our approach. Starting from 20cm, we found, when the length is less than 2.4cm, our method cannot detect surface defects. Note that we follow the specs entirely from the Microsoft Kinect website to set the parameters in Gazebo depth camera plugin.

5 Conclusion

In this paper, we proposed a front-end closed-loop control approach for 3D construction printing. We also explored the feasibility of using point cloud based defect defection for the construction size printing model. Comparing to other methods, our approach demonstrates a high performance defect detection method with a cost-effective sensor. As the experiments we showed, we believe that our proposed approach has great potential in the field of 3D construction printing.

Limitations and Discussions. An obvious limitation of our approach is that the detection accuracy will significantly decrease as a shrink of the printing model size. Generally, a smaller model will require higher performance and resolution of the 3D perception sensor to detect detail defects. Since the model size of 3D construction printing is large enough, and even a single layer could be easily captured by Kinect. Therefore, this is the main reason why we propose our method for defect detection in 3D construction printing.

Future Work. Our future work is aiming to bring out the real 3D printing product by using our approach. Furthermore, we believe that our work can ultimately help construction workers improve their safety and construction quality, help customers achieve their ideal designs, and reduce material waste to protect our environment.

Acknowledgment

The research is supported by NSF CPS program under CMMI-1932187.

References

[1] Velodyne lidar. https://velodynelidar.com/.

[2] Ouster lidar. https://ouster.com/.

[3] Dong Tian, Hideaki Ochimizu, Chen Feng, Robert Cohen, and Anthony Vetro. Geometric distortion metrics for point cloud compression. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3460–3464. IEEE, 2017.

[4] S Scott Crump. Apparatus and method for creating three-dimensional objects, June 9 1992. US Patent 5,121,329.

[5] Behrokh Khoshnevis. Automated construction by contour crafting—related robotics and information technologies. *Automation in construction*, 13(1):5–19, 2004.

[6] Behrokh Khoshnevis, Dooil Hwang, Ke-Thia Yao, and Zhenghao Yeh. Mega-scale fabrication by contour crafting. *International Journal of Industrial and Systems Engineering*, 1(3):301–320, 2006.

[7] Richard A Buswell, Rupert C Soar, Alistair GF Gibb, and A Thorpe. Freeform construction: mega-scale rapid manufacturing for construction. *Automation in construction*, 16(2):224–231, 2007.

[8] Steven J Keating, Julian C Leland, Levi Cai, and Neri Oxman. Toward site-specific and self-sufficient robotic fabrication on architectural scales. *Science Robotics*, 2(5):eaam8986, 2017.

[9] Winsun. http://www.winsun3d.com/En/Product/.

[10] Weiyi Lin, Hongyao Shen, Jianzhong Fu, and Senyang Wu. Online quality monitoring in material extrusion additive manufacturing processes based on laser scanning technology. *Precision Engineering*, 60:76–84, 2019.

[11] Zhen Liu, Suining Wu, Qun Wu, Chenggen Quan, and Yiming Ren. A novel stereo vision measurement system using both line scan camera and frame camera. *IEEE Transactions on Instrumentation and Measurement*, 68(10):3563–3575, 2018.

[12] Oliver Holzmond and Xiaodong Li. In situ real time defect detection of 3d printed parts. *Additive Manufacturing*, 17:135–142, 2017.

[13] Hongyao Shen, Weijun Sun, and Jianzhong Fu. Multi-view online vision detection based on robot fused deposit modeling 3d printing technology. *Rapid Prototyping Journal*, 2019.

[14] Tianjiao Wang, Tsz-Ho Kwok, Chi Zhou, and Scott Vader. In-situ droplet inspection and closed-loop control system using machine learning for liquid metal jet printing. *Journal of manufacturing systems*, 47:83–92, 2018.

[15] Wentai Zhang, Akash Mehta, Prathamesh S Desai, and C Higgs. Machine learning enabled powder spreading process map for metal additive manufacturing (am). In *Int. Solid Free Form Fabr. Symp. Austin, TX*, pages 1235–1249, 2017.

[16] Lu Lu, Jian Zheng, and Sandipan Mishra. A layer-to-layer model and feedback control of ink-jet 3-d printing. *IEEE/ASME Transactions on Mechatronics*, 20(3):1056–1068, 2014.

[17] Yijie Guo and Sandipan Mishra. A predictive control algorithm for layer-to-layer ink-jet 3d printing. In *2016 American Control Conference (ACC)*, pages 833–838. IEEE, 2016.

[18] Berk Altın, Zhi Wang, David J Hoelzle, and Kira Barton. Robust monotonically convergent spatial iterative learning control: Interval systems analysis via discrete fourier transform. *IEEE Transactions on Control Systems Technology*, 27(6):2470–2483, 2018.

[19] Patrick M Sammons, Michelle L Gegel, Douglas A Bristow, and Robert G Landers. Repetitive process control of additive manufacturing with application to laser metal deposition. *IEEE Transactions on Control Systems Technology*, 27(2):566–575, 2018.

[20] Azure kinect dk hardware specifications. https://docs.microsoft.com/en-us/azure/kinect-dk/hardware-specification.

[21] Ur10e collaborative industrial robot arm. https://www.universal-robots.com/products/ur10-robot/.

[22] Ros: The robot operation system. https://www.ros.org/.

[23] Moveit! https://moveit.ros.org/.