ID 49

Construction Site Top-view Generation Using Drone Imagery: The Automatic Stitching Algorithm Design and Application

Sisi Han[1], Yuhan Jiang[2]

[1] Marquette University, Milwaukee WI 53233, USA
[2] South Dakota State University, Brookings SD 57007, USA
yuhan.jiang@sdstate.edu

Abstract
This paper proposed an automatic stitching algorithm to process drone-captured top-view images (ortho-images) to generate a single frame high-resolution unified top-view image for construction site documentation. The initial step of the proposed algorithm is resizing adjacent ortho-images to the same scale. The next step is to find a common straight edge for merging the resized ortho-images. A vertical edge that closes to the right end of the left-image in left-right mode or a horizontal edge that closes to the bottom end of the upper-image in up-down mode is recommended. Then, merging adjacent ortho-images at the common edge. Stitching and aligning the corresponding elevation-maps at the common edge, if any. An automatic stitching tool was developed with comprehensive functions of automatic, semiautomatic, and manual stitching based on the stitching algorithm. Application results are presented and discussed in this paper, including grid stitching mode for large building sites and linear stitching mode for infrastructure projects. With the stitched ortho-image and elevation-map, the point cloud can be generated for 3D monitoring construction progress.

Keywords
Drone, Image stitching, Construction site, Documentation.

1. Introduction

Recent breakthroughs in micro quadrotor drones (or unmanned aerial vehicles, UAVs) have enabled dreams of drone photography and photogrammetry to get closer and closer to the reality of construction practices for construction site surveying (Shang & Shen, 2018; Siebert & Teizer, 2014), earthwork planning and monitoring (Haur et al., 2018; Nassar & Jung, 2012), and construction progress monitoring (Jacob-Loyola et al., 2021). Specifically, for construction site three-dimensional (3D) mapping, using drones with high-definition cameras for structure-from-motion (SfM) photogrammetry is the state-of-the-art approach for large construction site (Nassar & Jung, 2012; Vargo, 2020; Westoby et al., 2012). The commercial photogrammetry software packages, such as Autodesk ReCap Photo, can generate a two-dimensional (2D) orthophoto and a 3D point cloud for the scanned construction site (Jiang & Bai, 2021). However, it takes approximately one hour to process one hundred drone images in the SfM photogrammetry processing. As a result, “one processing day” is required for drone SfM photogrammetry-based soil measurement (Haur et al., 2018). When the contractor is only using drone imagery to document the construction project progress, using drone-captured 2D top-views of the construction site is enough. Then, with a series of top-views, project progress can be well monitored in building and infrastructure projects.

2. Data Acquisition

For architecture, engineering, and construction (AEC) applications and practices, "*DJI Phantom*" (Moon et al., 2019; Yang et al., 2018), "*DJI Inspire*" (Aguilar et al., 2019; Li & Lu, 2018), and "*DJI Mavic*" (Parket al., 2019) are the most popular ready-to-fly imaging drones. **Fig. 1** shows the components of a *DJI Phantom 4 Pro V2.0*, which is an easily controlled and portable quadcopter drone. It has a downward visionsystem and GPS for stable hovering at a planned position; the digital camera is mounted on a 3-axis (pitch,roll, yaw) gimbal to enhance the camera's stabilization (DJI, 2020). In this paper, ortho-imaging is definedas setting the gimbal's pitch-axis at –90° to make the camera lens face down to the construction site. Then,the drone captured images are top-views of the construction site with an approximate scale of ground sampling distance (*GSD*). Typically, *GSD* has the unit of cm/pixel, which means a pixel's length stands fora physical length of ground in centimeters.

As shown in **Fig. 2**, the *GSD* has an inverse relationship to drone flight height (altitude). To obtain high-resolution ortho-images, a low altitude is necessary. Then, for scanning the entire construction site, multiply stations are required for either a linear project or a larger area project; see the examples shown in **Fig. 3**. Next, stitching the overlapped ortho-images is required to generate a single frame high-resolution unified top-view of the construction site.

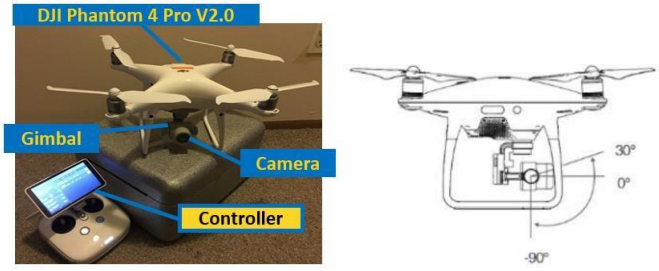

Fig. 1. Drone system, gimbal, and camera.

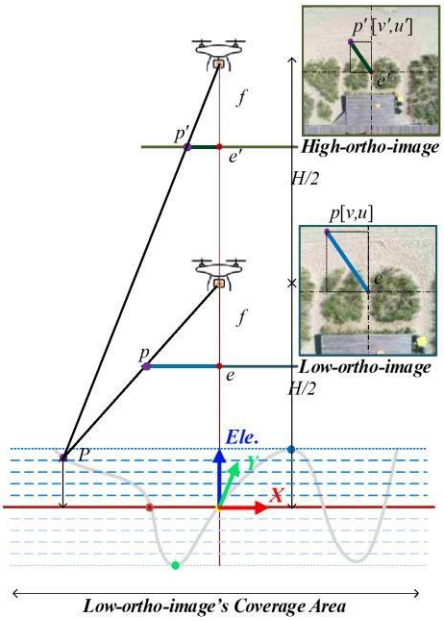

Fig. 2. Sketch of drone ortho-imaging in low-high altitudes**.**

Previous research (Jiang, 2020; Jiang & Bai, 2021) proposed an up-forward-down drone flight path to capture low-high image pairs (example shown in **Fig. 2**) on a construction site, where "forward" means the drone flight direction between two sequence stations; and "up" and "down" stand for the drone flight direction at each station, which is either moving up or down. When the construction site is prepared for a

linear infrastructural project like **Fig. 3**(a), where stations are set from the start point to the endpoint, then the left-right stitching mode is required to merge the adjacent results along the drone flight direction. When the construction site is prepared for a large building project like **Fig. 3**(b), where stations are planned in a grid style, the left-right mode is used in each row first, and then, the up-down mode is required for merging the stitched rows' results of images to cover the entire construction site.

Furthermore, the previously developed *PGMED* algorithm (Jiang, 2020; Jiang & Bai, 2021, 2020), *Fast-PGMED* algorithm (Han, 2020; Han et al., 2022) can quickly process low-high image pairs for construction site dense 3D reconstruction and elevation determination of each station covered area. The generated elevation-map has the same pixel coordinates as the low-image (dropped edges in four sides) at each station. Therefore, it would be great if a single frame high-resolution unified elevation-map could be created along with the ortho-image stitching.

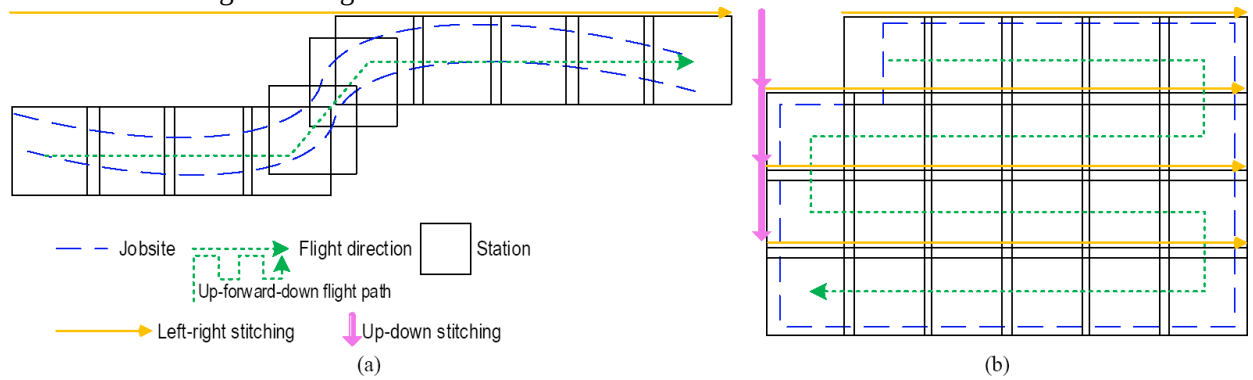

Fig. 3. Stitching modes: (a) linear infrastructure, (b) building.

3. Stitching Algorithm Design

This research aimed to optimized both stitching modes shown in **Fig. 3** to automatic level and also supported the ortho-images with different *GSDs*, which means ortho-images can be captured at different altitudes. **Fig. 4** demonstrates the processes of both left-right and up-down modes, where the matched keypoints are annotated as "×" with indication numbers in overlaps. The used keypoint is Scale-invariant Feature Transform (SIFT) keypoints, which is suitable for matching two images with different scales and orientations (D.G. Lowe, 1999; David G. Lowe, 2004).

The initial process is resizing adjacent ortho-images to have equal *GSD*. The linear scaling ratio can be calculated from the selected two pairs of matched SIFT keypoints like **Fig. 4**(a), where the largest distance is desired to guarantee the accuracy; it can also be automatically performed by selecting one keypoint at the beginning row and another at the ending row of the matched keypoint *DataFrame* (a type of data formatin *pandas*) (Pandas, 2020), in which keypoints are sorted by u-axis (along the horizontal direction of the ortho-image) for left-right mode and v-axis (along the vertical direction of the ortho-image) for up-down mode in increasing order like **Fig. 4** (a and c), respectively.

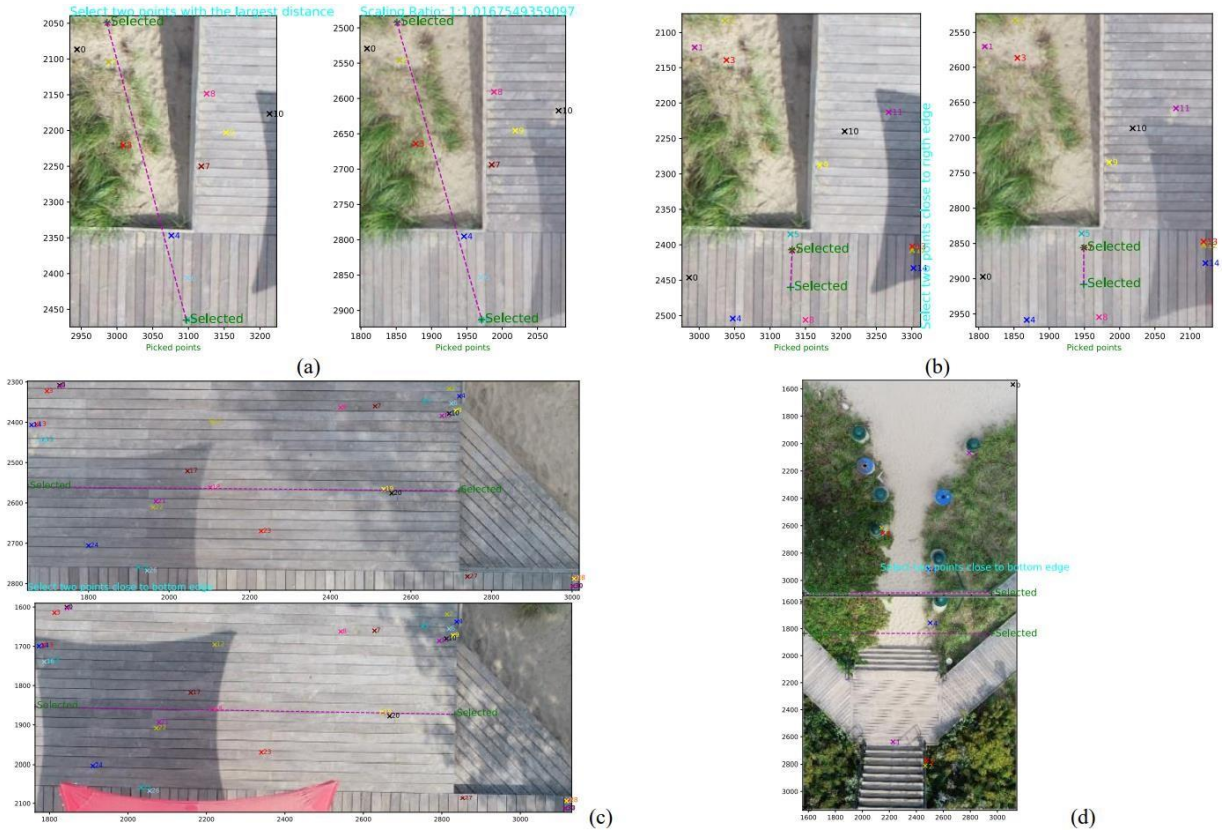

Fig. 4. Stitching processes: (a) scaling ratio, (b) left-right edge, (c) up-down edge, and (d) few SIFT keypoints**.**

The second process is determining a common straight edge for merging the resized ortho-images, preferably a vertical edge that closes to the right end of the left-image in left-right mode or a horizontal edge that closes to the bottom end of the upper-image in up-down mode. A suitable common edge can be determinedvia the proposed ***findCommonEdge***() algorithm shown in **Fig. 5**. For the left-right stitching in **Fig. 4**(b), the ***findCommonEdge***() algorithm uses the newly matched and sorted keypoint *DataFrame* of the resized ortho-images as the input. Moreover, when the number of matched SIFT keypoints is less than 30 to activate ***findCommonEdge***() or occurs with unsatisfied conditions, the selections of identical pixels can be manually performed like in **Fig. 4** (c and d).

findCommonEdge(*pointDataFrame*, *leftrightBool*): # *pointDataFrame* [u0, v0, u1, v1], matched keypoints; u0 and v0 refer the left or upper image
  for *condition*1 ***in*** *range*(1,50,1): # travel begins at the smallest value to guarantee the common edge in vertical or horizontal as much as possible
    for *i* ***in*** *range*(N − 1,0, −1): # travel begins at the largest value in u-axis (left-right mode) or v-axis (up-down mode); N, number of keypoints
      for *j* ***in*** *range*(i − 1,0, −1): # travel keypoints in order of u-axis (left-right mode) or v-axis (up-down mode)
        *index*0, *index* = *j*, *i* # index0, the first keypoint; index, the second keypoint; and then, two points determine a straight line
        u_{diff} = *pointDataFrame*[′**u0**′][*index*0] − *pointDataFrame*[′**u0**′][*index*] # difference between the current selected keypoints in u-axis
        v_{diff} = *pointDataFrame*[′**v0**′][*index*0] − *pointDataFrame*[′**v0**′][*index*] # difference between the current selected keypoints in v-axis
        *condition*0 = −50 × *condition*1 #adjustable condition0, guarantees the selected keypoints are not close to each other
        if ***abs***(u_{diff}) < *condition*1 ***and*** v_{diff} < *condition*0 ***and*** *leftrightBool* == *True*: # u_{diff} closer to zero, the line is more vertical
          return *index*0, *index* # negative v_{diff} guarantees first keypoint above the second keypoint
          break
        elif ***abs***(v_{diff}) < *condition*1 ***and*** u_{diff} < *condition*0 ***and*** *leftrightBool* == *False*: # v_{diff} closer to zero, the line is more horizontal
          return *index*0, *index* # guarantees first keypoint at the left of second selected keypoint; False, in case the up-down mode
          break

Fig. 5. Pseudocode of *findCommonEdge*() algorithm.

The additional processes to perform the ***stichingLeft&Right***() and ***stichingUp&Down***() algorithms include:

(1) Rotating both resized ortho-images to make the selected common edge vertical in left-right mode or horizontal in up-down mode, where both rotation centers are set at the first keypoints.
(2) Discarding the left-image's right part and the right-image's left part in left-right mode or discarding the up-image's bottom part and the down-image's upper part in up-down mode.
(3) Translating the right-image in v-axis to align with the left-image at the first keypoint in left-right mode or translating the bottom-image in u-axis to align with the upper-image at the first keypoint in up-down mode.
(4) Concatenating the left-right images or up-down images at the common edge.
Moreover, both ortho-images are padded in the above operations of rotation, translation, and concatenation to retain all ortho-images' information. The corresponding elevation-maps are automatically scaled and rotated along with ortho-images via the same keypoints and common edge. In addition, elevation-maps (8-bit grayscale image) are aligned to the same grayscale values at the common edge before concatenating them.
Furthermore, to automatically conduct stitching with multi-station, the ***stitchingLinear***() algorithm (see **Fig. 6**) is proposed to process a singular command list with the format of [0, '40Z', '40Y'], in which the "0" is used for activating ***stichingUp&Down*** **()** and "1" for ***stichingLeft&Right***( ), which returns the stitched results as '40Z_NEXT_Y'("40" in the file name indicate it is a 20-40-m ortho-image pair).

```
stitchingLinear(commandList = [0, '40Z', '40Y']):
  if len(commandList) == 2:
    return commandList[1]
  if commandList[0] == 0:
    baseName = commandList[1]
    nextNameList = commandList[2: ]
    for i in nextNameList:
      stichingUP&Down(baseName, i)
      baseName = baseName + "_NEXT_" + i
  elif commandList[0] == 1:
    baseName = commandList[1]
    nextNameList = commandList[2: ]
    for i in nextNameList:
      stichingLeft&Right(baseName, i)
      baseName = baseName + "_NEXT_" + i[2: ]
  return baseName
```

Fig. 6. Pseudocode of *stitchingLinear*() algorithm.

4. Application Results and Discussion

This research used the *PGMED* algorithm with low-high ortho-image pairs to determine construction sites' elevations. **Table 1** lists the properties of *PGMED*'s outputs of ortho-images and elevation-maps for each input of a pair of low-high ortho-images captured by a *DJI Phantom 4 Pro V2.0* over a target station at the low altitude $H/2$ and high altitude H. The scanned station X is referenced as 10-20 X or **20X**, in which 10 refers to the designated low altitude $H/2$=10m, 20 is the designated high altitude H=20m.

Table 1. Ortho-image and elevation-map at a station.

Properties	***10-20 ortho-image***	***20-40 ortho-image***
Drone captured image size	3648×4864-pixel	3648×4864-pixel
Assembled low-high ortho-image size	1824×1824-pixel	1824×1824-pixel
Generated ortho-image size	1568×1568-pixel	1632×1632-pixel
Generated elevation-map size	1568×1568-pixel	1632×1632-pixel
Approximate *GSD*	0.54 cm/pixel	1.08 cm/pixel
Covered construction site area	8.5×8.5 m^2	17.6×17.6 m^2

Measurable elevations	[-5 m,5 m]	[-10m, 10m]
Elevation-map value	[0,255]	[0,255]
Elevation-map interval	0.0392-m	0.0784-m
Elevation-map system error	0.0196-m	0.0392-m

Note: see the detailed definitions and values of the listed properties in (Jiang & Bai, 2021)

Since the captured ortho-images have GPS coordinates, their related directions can be easily determined. The four ortho-images, shown in **Fig. 7**(a), have an approximate 2×2 grid relationship. Then, the stitching command list can be written as [[1, '40Z', '40Y'], [1, '40S', '40V']] for the ***stitchingGrid***() algorithm shown in **Fig. 8**, where each singular command list is a row of **Fig. 7**(a). At the end of the process, the ***stitchingGrid***() algorithmcreates a command list [0, '40Z_NEXT_Y', '40S_NEXT_V'] for applying the up-down stitching of the two stitched rows by ***stitchingLinear***() and returns the 2×2 stitched **40Z_NEXT_Y_NEXT_S_NEXT_V**, as shown in **Fig. 7**(d).

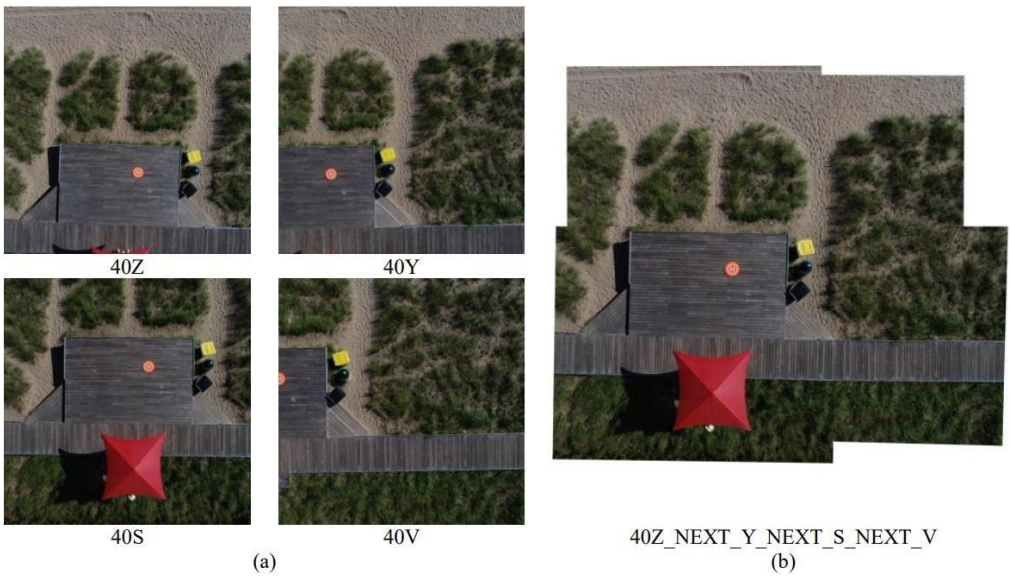

Fig. 7. Stitching modes: (a) 2×2 grid, and (b) stitched ortho-image.

```
stitchingGrid(commandList = [[1,'40Z','40Y'],[1,'40S','40V']]):
  colNameList = [0]
  for rowlist in commandList:
    colNameList.  append(stitchingLinear(rowlist))
  stitchingLinear(colNameList)
```

Fig. 8. Pseudocode of *stitchingGrid*() algorithm.

Three pairs of ortho-images and elevation-maps of **40CA**, **40CH**, and **40CI** ("40" indicates they are 20-40-m ortho-image pairs) with the image size of 1632×1632-pixel were collected from (Jiang & Bai, 2021); an ortho-image and elevation-map pair of **40CJ** with size of 1568×1568-pixels was collected from (Han, 2020). These four pairs of ortho-images and elevation-maps were automatically stitched with ***stitchingGridColThenRow***() algorithm (shown in **Fig. 9**) via the command [[0,'40CJ','40CI'], [0,'40CH'], [0,'40CA']], which called ***stitchingUp&Down***() to merge **40CJ** and **40CI** first. Then, the merged results of **40CJ_NEXT_CI** were stitched with **40CH** and **40CA** in sequence via ***stitchingLeft&Right***().

```
stitchingGridColThenRow(commandList = [[0, '40CJ', '40CI'], [0, '40CH'], [0, '40CA']]):
  rowNameList = [1]
  for collist in commandList:
    rowNameList.append(stitchingLinear(collist))
  stitchingLinear(rowNameList)
```

Fig. 9. Pseudocode of *stitchingGridColThenRow*() algorithm.

Additionally, the developed automatic stitching tool has comprehensive functions and graphic user interfaces (GUIs) for automatic, semiautomatic, and manual stitching. In process command [[0, '40CJ', '40CI'], [0, '40CH'], [0, '40CA']], the automated stitching was activated in the up-down stitching of **40CJ** and **40CI**, while semiautomatic stitching and manual stitching were used in the remaining two left-right stitching because the matched SIFT keypoints were less than the thresholds (10 for determining scaling ratio and 30 for determining common straight edge) to activate automated stitching. In **Fig. 10**(a), common keypoints of 19 and 18 were picked to determine the common edge via typing them into the command line. In **Fig. 10**(b), the common keypoints of 6 and 7 were discarded because the determined straight line occurred on the edge of the wooden platform and would produce errors in elevation alignment. Thus, the keypoint 0 was set as the starting point, and the ending points were picked along the direction of the chink between two boards via cursor in the final. The final stitched results of ortho-image and elevation-map are shown in **Fig. 11**, where a wooden path links two stairs and a platform.

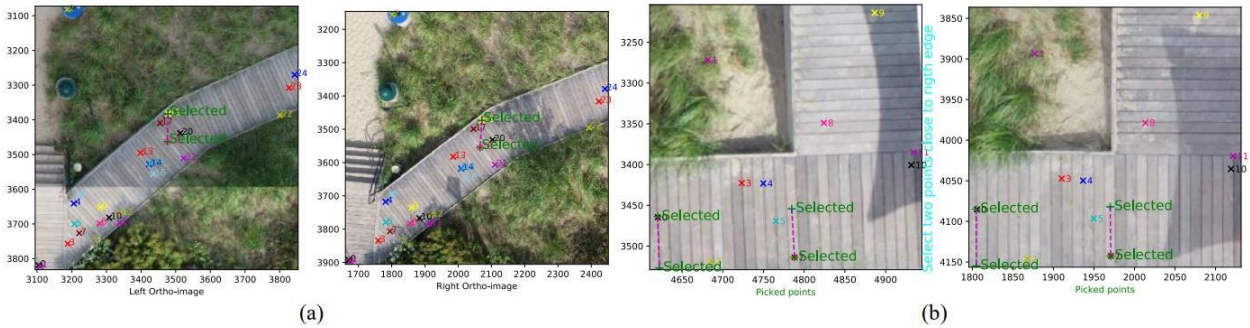

Fig. 10. Stitching examples: (a) semiautomatic left-right stitching and (b) manual left-right stitching.

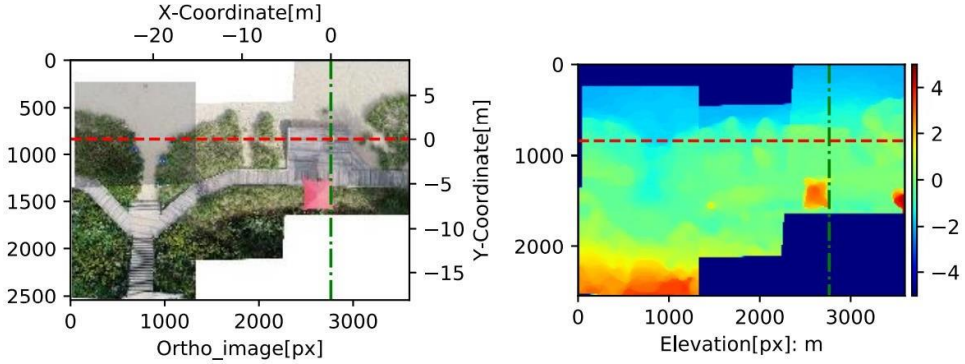

Fig. 11. Stitched ortho-image, elevation-map.

Furthermore, the *GSD* of 1.04 cm/pixel was determined for the stitched ortho-image and elevation-map via detecting the placed drone landing pad, same as the examples shown in (Han et al., 2022; Jiang & Bai, 2021). The determined *GSD* is close to the approximate value listed in **Table 1**, which means the drone's altitudes were not at 20-40m to the ground. Thus, *GSD* adjustment is necessary for the 3D point cloud generation, and the stitched ortho-image and elevation-map converted 3D point cloud is shown in **Fig. 12**.

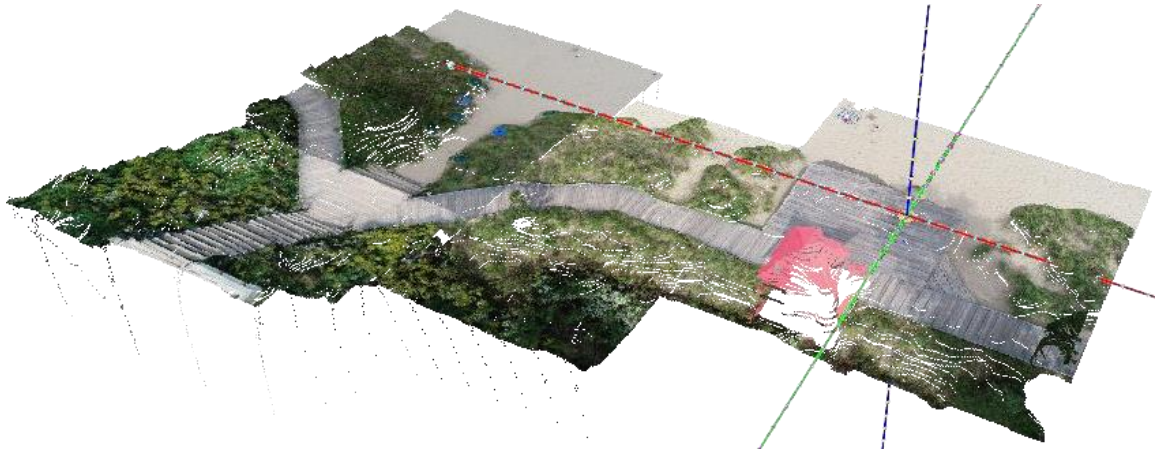

Fig. 12. 3D point cloud.

5. Conclusion

This paper proposed an automatic stitching algorithm to process drone captured top-view images (ortho-images) to generate a single frame high-resolution unified top-view image for construction site documentation with the following steps:
(1) The initial step of the proposed stitching algorithm is resizing adjacent ortho-images to have equal scale.
(2) Next, a common straight edge for merging the resized ortho-images is determined. A vertical edge closes to the right end of the left-image in left-right mode, or a horizontal edge that closes to the bottom end of the upper-image in up-down mode is recommended.
(3) Then, adjacent ortho-images are merged at the common edge.
The automatic stitching tool was developed with comprehensive functions and graphic user interfaces (GUIs) for automatic, semiautomatic, and manual stitching based on the proposed algorithm. Applications of 2×2 grid stitching and the mixed stitching were tested. The corresponding elevation-maps were aligned and stitched along with the ortho-image stitching.
The proposed drone imaging approach can be applied to document construction sites during the preconstruction, construction, and constructed phases. The coverage can be extended via automated image stitching, and the coordinate conversion and orientation alignment can be completed via setting a single landing pad as the ground control point. With the *PGMED*/*Fast-PGMED* algorithm, the elevation-map and point cloud could be created. Moreover, with deep learning-based segmentation approaches (Jiang et al., 2020, 2022), construction site objects could be easily extracted for as-constructed modeling.

References

Aguilar, R., Noel, M. F., & Ramos, L. F. (2019). Integration of reverse engineering and non-linear numerical analysis for the seismic assessment of historical adobe buildings. *Automation in Construction*, *98*, 1–15. https://doi.org/10.1016/j.autcon.2018.11.010

DJI. (2020). *Phantom 4 Pro V2.0*. https://www.dji.com/phantom-4-pro-v2/specs

Han, S. (2020). *Using Drone-Based Fully Convolutional Network for the Determination of Construction Site Elevations* [Marquette University]. https://epublications.marquette.edu/theses_open/610

Han, S., Jiang, Y., & Bai, Y. (2022). Fast-PGMED: Fast and Dense Elevation Determination for Earthwork Using Drone and Deep Learning. *Journal of Construction Engineering and Management*, *148*(4). https://doi.org/10.1061/(ASCE)CO.1943-7862.0002256

Haur, C. J., Kuo, L. S., Fu, C. P., Hsu, Y. L., & Heng, C. Da. (2018). Feasibility Study on UAV-assisted Construction Surplus Soil Tracking Control and Management Technique. *IOP Conference Series: Materials Science and Engineering*, *301*, 012145. https://doi.org/10.1088/1757-899X/301/1/012145

Jacob-Loyola, N., Muñoz-La Rivera, F., Herrera, R. F., & Atencio, E. (2021). Unmanned Aerial Vehicles

(UAVs) for Physical Progress Monitoring of Construction. *Sensors*, *21*(12), 4227. https://doi.org/10.3390/s21124227

Jiang, Y. (2020). *Determination of Elevations for Excavation Operations Using Drone Technologies* [Marquette University]. https://epublications.marquette.edu/dissertations_mu/988/

Jiang, Y., & Bai, Y. (2021). Low–High Orthoimage Pairs-Based 3D Reconstruction for Elevation Determination Using Drone. *Journal of Construction Engineering and Management*, *147*(9), 04021097. https://doi.org/10.1061/(ASCE)CO.1943-7862.0002067

Jiang, Y., & Bai, Y. (2020). Determination of Construction Site Elevations Using Drone Technology. *Construction Research Congress 2020*, 296–305. https://doi.org/10.1061/9780784482865.032

Jiang, Y., Bai, Y., & Han, S. (2020). Determining Ground Elevations Covered by Vegetation on Construction Sites Using Drone-Based Orthoimage and Convolutional Neural Network. *Journal of Computing in Civil Engineering*, *34*(6), 04020049. https://doi.org/10.1061/(ASCE)CP.1943- 5487.0000930

Jiang, Y., Han, S., & Bai, Y. (2022). Construction Site Segmentation Using Drone-Based Ortho-Image and Convolutional Encoder-Decoder Network Model. *Construction Research Congress 2022*, 1096–1105. https://doi.org/10.1061/9780784483961.115

Li, D., & Lu, M. (2018). Integrating geometric models, site images and GIS based on Google Earth and Keyhole Markup Language. *Automation in Construction*, *89*, 317–331. https://doi.org/10.1016/j.autcon.2018.02.002

Lowe, D.G. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1150–1157 vol.2. https://doi.org/10.1109/ICCV.1999.790410

Lowe, David G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journalof Computer Vision*, *60*(2), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

Moon, D., Chung, S., Kwon, S., Seo, J., & Shin, J. (2019). Comparison and utilization of point cloud generated from photogrammetry and laser scanning: 3D world model for smart heavy equipment planning. *Automation in Construction*, *98*, 322–331. https://doi.org/10.1016/j.autcon.2018.07.020

Nassar, K., & Jung, Y.-H. (2012). Structure-From-Motion Approach to the Reconstruction of Surfaces forEarthwork Planning. *Journal of Construction Engineering and Project Management*, *2*(3), 1–7. https://doi.org/10.6106/JCEPM.2012.2.3.001

Pandas. (2020). *pandas.DataFrame*. https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html

Park, J., Kim, P., Cho, Y. K., & Kang, J. (2019). Framework for automated registration of UAV and UGVpoint clouds using local features in images. *Automation in Construction*, *98*, 175–182. https://doi.org/10.1016/j.autcon.2018.11.024

Shang, Z., & Shen, Z. (2018). Real-Time 3D Reconstruction on Construction Site Using Visual SLAM andUAV. *Construction Research Congress 2018*, 305–315. https://doi.org/10.1061/9780784481264.030 Siebert, S., & Teizer, J. (2014). Mobile 3D mapping for surveying earthwork projects using an UnmannedAerial Vehicle (UAV) system. *Automation in Construction*, *41*, 1–14. https://doi.org/10.1016/j.autcon.2014.01.004

Vargo, L. J. (2020). Landscape surveys from the sky. *Nature Reviews Earth & Environment*, *1*(2), 87–87. https://doi.org/10.1038/s43017-020-0023-4

Westoby, M. J., Brasington, J., Glasser, N. F., Hambrey, M. J., & Reynolds, J. M. (2012). 'Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications. *Geomorphology*, *179*, 300–314. https://doi.org/10.1016/j.geomorph.2012.08.021

Yang, C.-H., Tsai, M.-H., Kang, S.-C., & Hung, C.-Y. (2018). UAV path planning method for digital terrainmodel reconstruction – A debris fan example. *Automation in Construction*, *93*, 214–230. https://doi.org/10.1016/j.autcon.2018.05.024