sigradi 2021 designing possibilities ubiquitous conference

A Machine Learning Approach to Translate Graph Representations into Conceptual Massing Models

Kaan Karabagli [1], Mustafa Koc [2], Prithwish Basu [3], Imdat As [1]

[1] Istanbul Technical University, Turkey
kaankarabagli@gmail.com
[2] TÜBİTAK, Turkey
koc.mustafa@outlook.com
[3] Raytheon BBN Technologies, USA
prithwish.basu@raytheon.com
imdatas@gmail.com

Abstract: Machine learning (ML) has popular applications in domains involving image, video, text and voice. However, in architecture, image-based ML systems face challenges capturing the complexity of three-dimensional space. In this paper, we leverage a graph-based ML system that can capture the inherent topology of architectural conceptual designs and identify high-performing latent patterns within such designs. In particular, our goal is to translate architectural graph data into three-dimensional massing models. We are building on our prior ML work, where we, a. discovered latent topological features, b. composed building blocks into new designs, c. evaluated their feasibility, and d. explored Generative Adversarial (Neural) Networks (GAN)-generated design variations. We trained the ML system with architectural design data that we gathered from an online architectural design competition platform, translated them into machine-readable graph representations, and identified their essential subgraphs to develop novel compositions. In this paper, we explore how these novel designs (outputted in graph form), can be translated into three-dimensional architectural form. We present an ML approach to turn graph representations into functional volumetric massing models. The ultimate goal of the study is to develop an end-to-end pipeline to generate architectural design - from a graph representation to a fully developed conceptual proxy of a designed product. The research question is promising in automating conceptual design, and we believe the outcome can be relevant to other design disciplines as well.

Keywords: Architectural design, machine learning, conceptual design, deep learning, artificial intelligence

1 Introduction

Architecture is a complex discipline. The digital revolution brought significant developments to the architectural design practice. Currently there are various methods that incorporate ML in the architectural design process, but these are typically based on generating 2D floor plans.

Machine learning offers significant potential to provide apposite, timely, and scalable solutions and new ways to study design processes. In this paper, we illustrate a procedure that extends the use of ML in the three-dimensional space by generating conceptual massing models, which can be used in the early stages of the design process.

First, we take a graph representation of an architectural building and explore how it can be translated into various floor plans; second, we choose the functional floorplans generated though HouseGAN++ and use them in the next step to generate a three-dimensional volumetric outcome. Finally, we stack the conceptual massing models to produce the final design model. This process enables ML to turn graph representations into functional volumetric mass models and makes an initial attempt to automate the conceptual design phase.

2 Background

ML-based systems have been used by creative industries in recent years. Tools like RunwayML allow creatives to incorporate AI in their works. Even though these types of tools are not specifically developed for architecture, they allow designers to input their data directly into the software and perform various tasks. These emergent tools are lowering the barriers to utilize AI in architecture, requiring no coding experience and running complex tasks without extensive hardware requirements (https://docubase.mit.edu/tools/runwayml/).

Besides RunwayML there are several other software tools that use ML, such as “Syntactic”, “Magnetizing Floor Plan Generator” and “Termite Nest”. (https://www.food4rhino.com/en), to mention a few. These prototype plug-ins enable designers to set basic parameters in their design process and can be considered as an exploration in using algorithms to solve certain design tasks. However, they do not produce a three-dimensional outcome.

In this paper, we build on our prior work (“Artificial Intelligence in Architecture: Generating Conceptual Design via Deep Learning” (As, Pal, & Basu, 2018)). In this previous study, we had presented a function-driven deep learning approach to generate conceptual designs by training a deep neural network. The training data consisted of Building Information Models (BIM) of residential building units that were translated into graph representations. The

deep neural network (DNN) was able to discover high performing latent sub-graphs and assemble them into new conceptual design compositions (Fig.1).

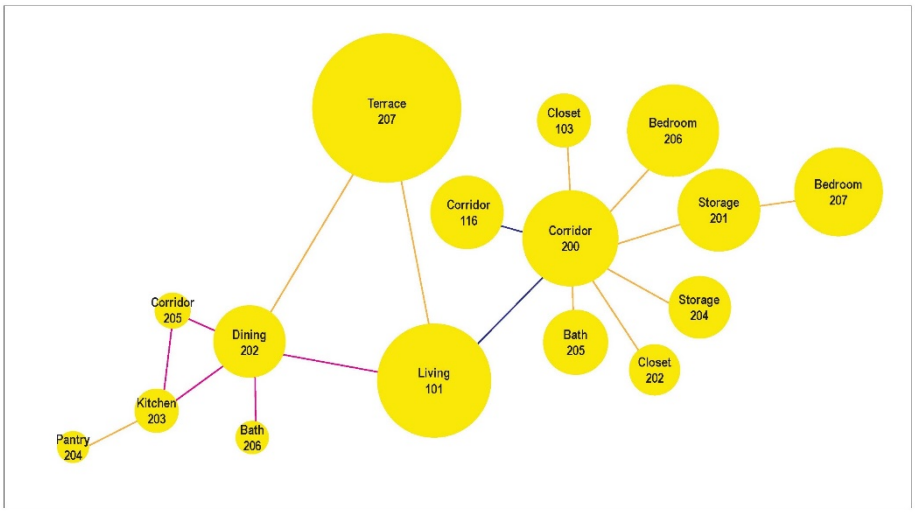

Figure 1. Composition of high-performing subgraphs generated from two homes. Figure adapted from previous publication (As, Pal, & Basu, 2018) with permission.

In this study, our goal is to translate these subgraphs into three-dimensional massing models. We surveyed existing machine learning software developed specifically for the architectural design space. We used HouseGAN++ (2020) to transform graphs into floorplans and then processed the images to generate 3D massing models by the Floorplan-to-Blender3D algorithm (2021).

However, the strategy for this study is to have a one single automated process used within a single software. The HouseGAN++ and FloorplanToBlender algorithms that are currently used in this study require certain tasks that require manual applications. The current HouseGAN++ successfully generates floorplans, but these are all separate floorplans for each individual floor levels and the FloorplanToBlender successfully generates the 3D models of each floorplan separately after processing the floorplan images accordingly. The process of choosing the matching floor levels and stacking them into one single 3D massing model is applied manually based on architectural knowledge and decision making. Our goal is to edit and evolve the HouseGAN++ and FloorplanToBlender we currently use and have a fully automated new algorithm that could generate a multilevel 3D massing model in a single process.

3 Methods and Tools

Traditionally architects would use bubble diagrams in their early stages of conceptual design work to develop a given building program into a floor plan, and then into a three-dimensional architectural work. (Fig. 2)

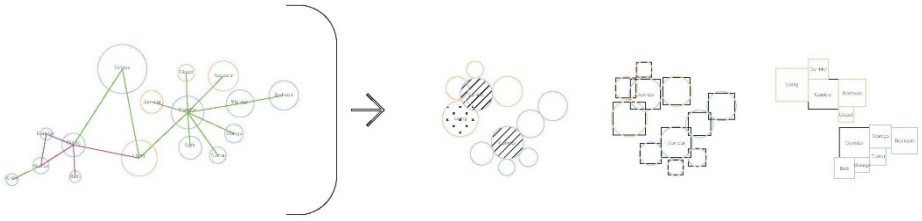

Figure 2. Traditionally architects translating a given program into a bubble-diagram and then into a floorplan.

In this study, we demonstrate a protocol through which this age-old modus operandi can be potentially automated, i.e., architectural graph data translated into three-dimensional conceptual massing models. We expand on our prior ML work, where we discovered latent topological features, i.e., essential building blocks, in architectural building data, composed them into new designs, evaluated their feasibility, and explored GANs to generate unprecedented new designs.

We first utilized HouseGAN++ to transform a graph into a floorplan, then used MatLAB to extract the shape features such as rooms, walls, and doors in the floorplan image to a text file – in order to extrude them into a 3D model. Afterwards, we used the Floorplan-to Blender3D procedure that takes these floorplan images and generates a three-dimensional conceptual massing model.

HouseGAN++ is a form of a generative adversarial network (GAN). GANs are a promising development in deep learning. They can be used in a wide range of applications, e.g., generating unprecedented imagery from training data. HouseGAN++ was trained by a database of 117,587 floorplans collected from the LIFULL HOME database that consists of node diagrams (Nauata, et al., 2021). The floorplans were transformed to graphs by convolutional message passing neural networks (Conv-CPN). Thus HouseGAN++ is able to generate a variety of floorplans from bare input graphs. Here, HouseGAN++ is

used to convert a graph into a floorplan unlike the earlier use of GAN in our prior work (As, Pal, & Basu, 2018), where it was used to generate the graph itself from training data.

Conv-MPN is built on a relational graph structure, that can read graph representations consisting of data in the form of nodes and edges. Conv-MPNs use latent 3D volume to represent features, and matrix multiplications with convolutions for the message passing; this differs from standard graph neural networks (GNN) that use 1D vector to numerically encode geometry information.

Conv-MPN takes a graph with an explicit spatial embedding, represents a node as feature volume and performs convolutions in the spatial Feature volume is converted into a room segmentation mask by a shared three-layer convolutional neural network (CNN) then passed to a discriminator during the training process. The discriminator performs a sequence of operations in reverse order.

We passed the results of the House-GAN++ algorithm into MATLAB to extract various shape features in the floorplans -- in order to build a 3D model. Images are processed with watershed and Harris corner detection, and distance transform algorithms that extract, for example, rooms, walls, door features, etc. (Grebtsew, 2021). The watershed algorithm is used to separate different objects in images by treating pixels as a local topography and floods basins from the markers until basins attributed to different markers meet on watershed lines. In theory, the grayscale image is viewed as a topographic surface where high intensity denotes hills and low intensity stands for valleys (OpenCV, 2021). Then surroundings of all local minima are detected until they merge with each other. The pixels where merging occurs give the segmentation results. After watershed segmentation, images are processed with Harris corner detection algorithm to detect corner points of rooms and walls to extract the features. The grayscale of the original image is applied with gaussian filter to smooth the noises for each pixel in the grayscale image. A 3x3 window is considered to compute corner strength function. For pixels that are over the threshold and are local maxima within a certain window, a feature descriptor is computed (Tyagi, 2019). After the Harris corner detection, the grayscale image is processed for computing Euclidean *distance transform*. For each pixel in the image, the *distance transform* assigns a number that is the distance between that pixel and the nearest nonzero pixel of the image. The result of the transform shows the distance to the closest boundary from each point. After extracting the shape features from the floorplan images, these features are used in Grebtsew's algorithm for creating 3D massing models of floorplans (Grebtsew, 2021).

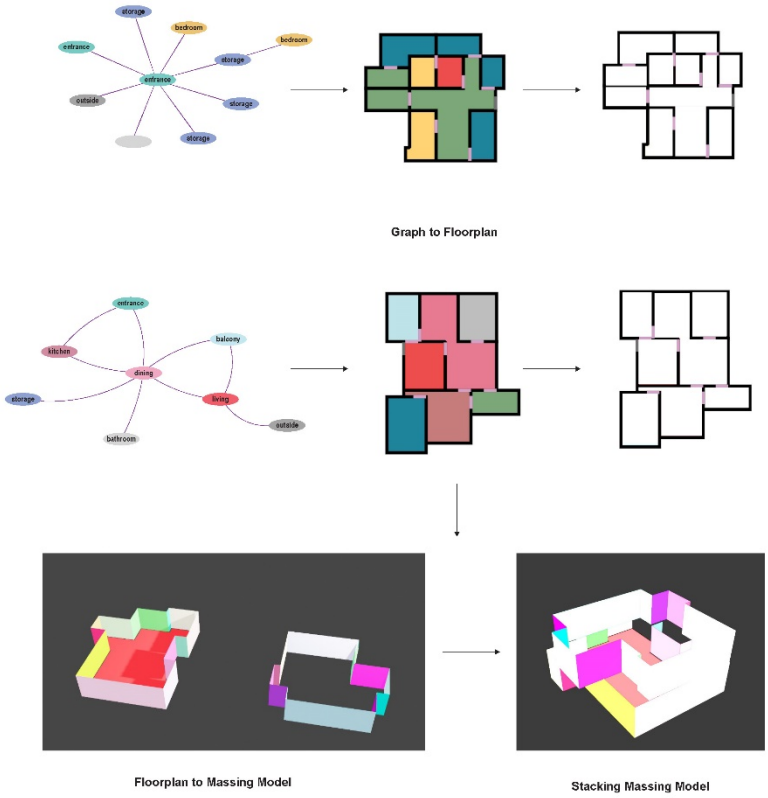

Figure 3. Overall schema of the study.

4 Case Study

In this case study we used some of the high performing subgraphs produced in the earlier work mentioned above (As, Pal, & Basu, 2018) and translated them into 3D conceptual massing models. In particular, the graphs we started can be seen in Fig.1. The workflow is as follows (Fig.3):

a) Picking a high-performing subgraph
b) Use subgraph as input data to HouseGAN++
c) Extrude floorplans to 3D
d) Stack different floorplan levels to produce final conceptual massing model

a. Picking a high-performing subgraph

For this study we picked the subgraphs shown in Fig.5 which was part of the larger graph of a house (fig.2) from the study (As, Pal, & Basu, 2018). The reason we picked a set of these particular subgraphs is because it consisted of a rich subset of the building program.

b. Use subgraph as input data to HouseGAN++

We used the subgraph as input data to the House-GAN++ application to translate the graph into a floorplan (Nauata, et al., 2021). In the graphs, nodes encode rooms with room type, and edges encode spatial adjacency. The algorithm depends on Conv-MPN, which replaces a latent vector in standard message passing networks with 3D volume for feature representation and fully connected layers with convolutions for the message encoding (Zhang, Nauata, & Furukawa, 2020). In the House-GAN++ application, edges carry features of door generation, each node and edge takes a 2D segmentation mask with an associated loss, and Conv-MPN feature pooling is reformulated to allow exchanging features between nodes and edges (Nauata, et al., 2021). They used ground-truth conditional training as a training process that picks a ground-truth layout at random, initializes a relational graph then specifies a ground-truth segmentation mask as input condition for each room/door with 50% chance to improve design by passing previous generated layouts as input condition (Nauata, et al., 2021).

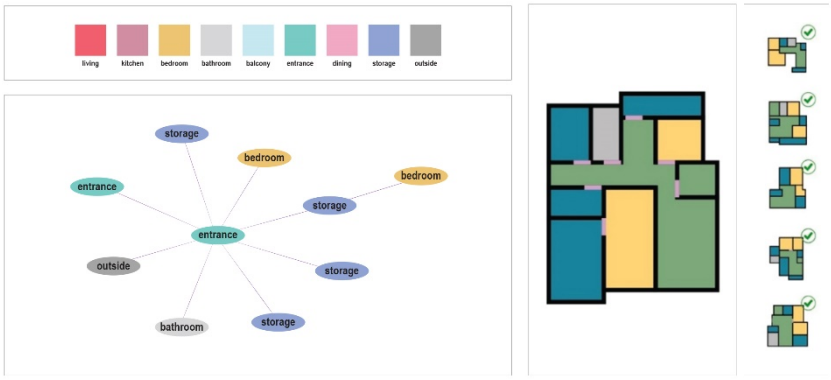

Figure 4. Floorplans generated from node graph example 1.

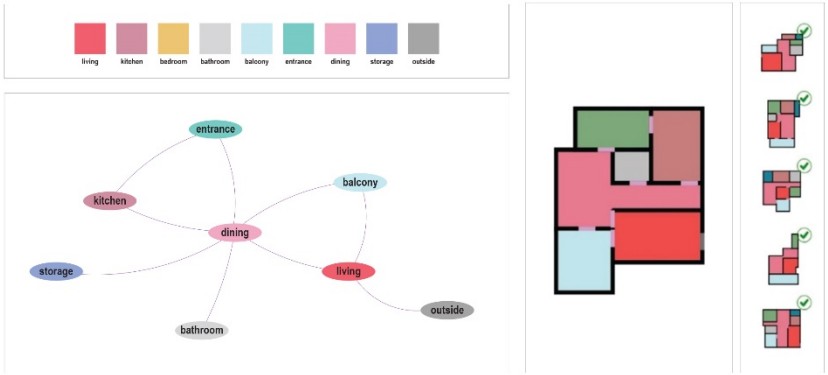

Figure 5. Floorplans generated from node graph example 2.

In this process, we chose the most functional floorplan that has been generated through HouseGAN++, and used it in the next step to generate a three-dimensional volumetric outcome.

c. Extrude floorplans into 3D

The generated floorplan images are processed to detect walls, doors, windows, and rooms to create a 3D massing model. We selected the image to 3D model algorithm on Github and processed the floorplan images in Matlab to make them machine-readable (Grebtsew, 2021). Floorplan images are processed for shape and room detection to create a text file that stores shape features such as detected shapes' pixel locations. Then, these text files are used to create a 3D massing model in the Blender3D application. Images are processed with the watershed and Harris corner detection algorithms, and distance transform to detect rooms and walls with respect to their shapes. After detecting shapes from images, their features are saved to a text file to create a 3D model. Grebtsew's algorithm uses the shape features to create 3D massing models in the Blender3D application. It uses a server that contains Swagger API, to convert images to 3D models by floor to blender (FTBL) library that is created by the author (Grebtsew, 2021).

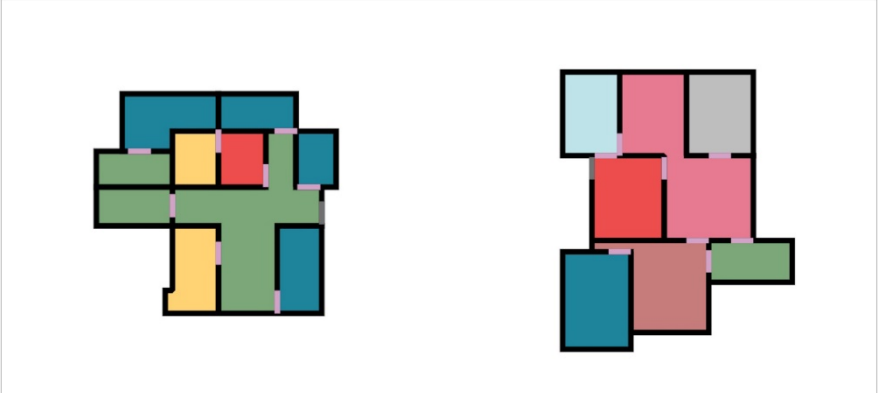

Figure 6. Floorplans for different floor levels of same building.

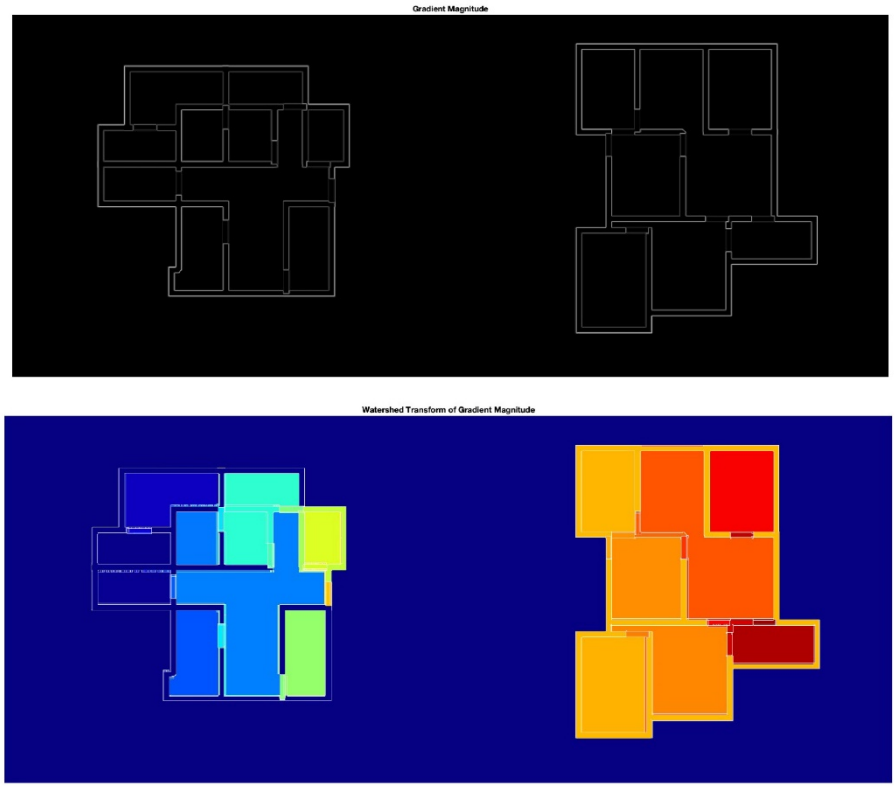

Figure 7. Image processing for detecting shape features and transforming floorplans to create a 3D model.

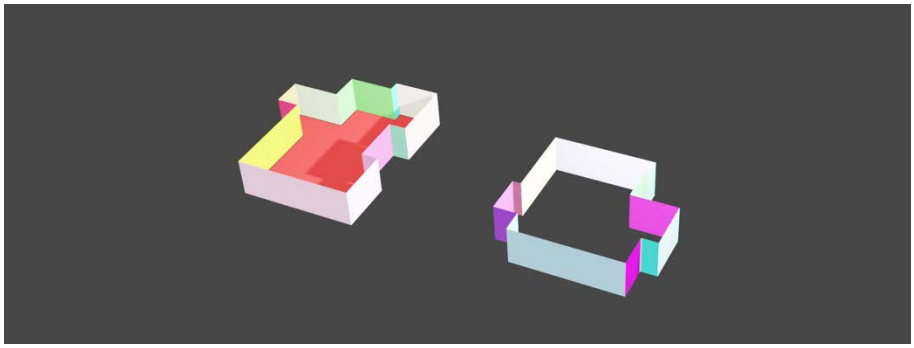

Figure 8. 3D massing models example based on two subgraphs.

d. Stack different floorplan levels to produce conceptual massing model

Algorithmically stacking the 3D models of floorplans is a topic of active research for us. Currently, the stacking process takes place manually using based on architectural knowledge and decision making.

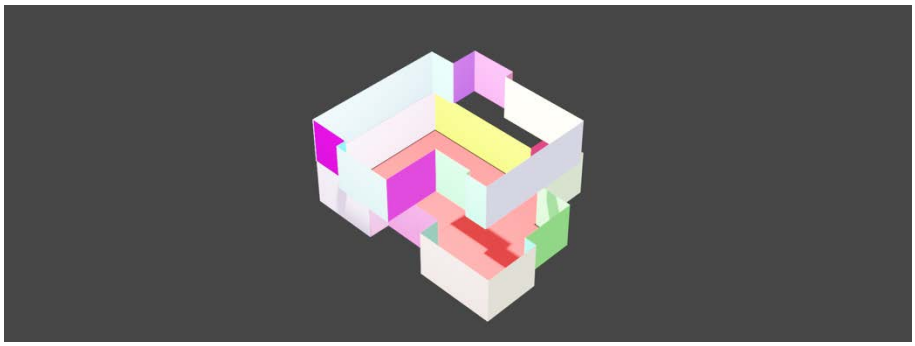

Figure 9. Stacking of massing models to produce final conceptual massing model.

5 Conclusion

The aim of this paper is to illustrate a new procedure for architects to create design alternatives from graph representations. The ML system can create infinite design alternatives. We build upon the previous study of As et al. and developed a method to convert a graph into a 3D model. We first translated the graph into a floor plan, whose shape features, e.g., walls, doors, etc. are automatically detected. Afterwards we generated a 3D model of the floorplan in Blender3D.

There are limitations to the protocol we present in this paper. Firstly, it can only deal with single floor buildings. Therefore, buildings with multiple floors have to be manually stacked. Secondly, the tool can only create box shaped floorplans. We believe we can address these limitations in future iterations of the protocol. We are continuing to work on stacking multiple floor plans for consecutive floors while obeying the “vertical” adjacency relationships such as staircase, elevator, etc., in a complete multi-floor building graph.

The potential of automating the conversion of graphs into conceptual massing models could offer vast opportunities for the early stages of the design process. While the ML system generates basic conceptual massing models the designer could craft these into highly developed complete designs in the future.

“This document does not contain technology or technical data controlled under either the US International Traffic in Arms Regulations or the US Export Administration Regulations.”

References

As, İ., Pal, S., & Basu, P. (2018). Artificial Intelligence in Architecture: Generating Conceptual Design via Deep Learning. International Journal of Architectural Computing, 16(4), 306-327.

Grebtsew. (2021, May 7). FloorplanToBlender3d. Retrieved from Github: https://github.com/grebtsew/FloorplanToBlender3d

Nauata, N., Hosseini, S., Chang, K.-H., Chu, H., Cheng, C.-Y., & Furukawa, Y. (2021, March 3). House-GAN++: Generative Adversarial Layout Refinement Networks. Retrieved from Cornell University: https://arxiv.org/abs/2103.02574

OpenCV. (2021, April 2). Image Segmentation with Watershed Algorithm. Retrieved from Open Source Computer Vision: https://docs.opencv.org/4.5.2/d3/db4/tutorial_py_watershed.html

Tyagi, D. (2019, March 16). Introduction to Harris Corner Detector. Retrieved from Data Breach: https://medium.com/data-breach/introduction-to-harris-corner-detector-32a88850b3f6

Zhang, F., Nauata, N., & Furukawa, Y. (2020, March 31). Conv-MPN: Convolutional Message Passing Neural Network for Structured

Outdoor Architecture Reconstruction. Cornell University: https://arxiv.org/abs/1912.01756

Solemma. (2020). ClimateStudio (version 2.4) [Computer software]. https://www.solemma.com/cs-trial

OpenCV. (2021, April 2). Image Segmentation with Watershed Algorithm. Retrieved from Open Source Computer Vision: https://docs.opencv.org/4.5.2/d3/db4/tutorial_py_watershed.html

Tyagi, D. (2019, March 16). Introduction to Harris Corner Detector. Retrieved from Data Breach: https://medium.com/data-breach/introduction-to-harris-corner-detector-32a88850b3f6