Quick-Pro-Build: A Web-based Approach for Quick Procedural 3D Reconstructions of Buildings

B. Bohlender[1,2], M. Mühlhäuser[1], and A. Sanchez Guinea[1]

[1]TU Darmstadt, Germany
[2]Coconut Capital GmbH, Germany

Abstract

We present Quick-Pro-Build, a web-based approach for quick procedural 3D reconstruction of buildings. Our approach allows users to quickly and easily create realistic 3D models using two integrated reference views: street view and satellite view. We introduce a novel conditional and stochastic shape grammar to represent the procedural models based on the well-established CGA shape grammar. Based on our grammar and user interface, we propose 3 modalities for procedural modeling: 1) model from scratch, 2) copy, paste, and adapt, and 3) summarize, select and adapt. The third modality enables users to model a building by summarizing similar models into an architectural style description, selecting a model from the style description, and adapting it to the target building. Summarizing and selecting allows the third modality to be the most efficient option when modeling a building with a style similar to existing buildings. The third modality is enabled by a novel algorithm that can find and combine similarities from procedural models into a style description and allows learning the preference of the users for one model inside the style description.

CCS Concepts

• ***Computing methodologies*** → ***Computer graphics;***

1. Introduction

Despite the rising demand for high-quality 3D reconstructions of urban environments, current solutions either lack detail (e.g., OpenStreetMap), are only available for a limited area (e.g., 3DCityDB [YNK*18]), produce coarse results (e.g., [YC16]), or require large amounts of data (e.g., [KFWM17]). Furthermore, approaches for detailed manual reconstruction are still not particularly efficient (e.g., [LWW08], [KK12], [ARB07]) and typically not accessible. Meanwhile, accessible web-based procedural model viewers exist but lack editing capabilities (e.g., [LGMS16], [NDBP15]).

In this paper, we propose Quick-Pro-Build, a web-based approach for quick interactive procedural 3D reconstruction of buildings. Our approach combines an intuitive graphical user interface, 3 modalities for efficient urban modeling, and the accessibility of the web with a novel stochastic procedural modeling language inspired by the CGA language [MWH*06]. The graphical user interface builds on paradigms, such as copy and paste [ARB07], 3D manipulators [KK12], and *local edits* [LWW08]. The interface integrates satellite and street view imagery [SSH17] as reference and is available as a web application. The 3 modalities enable efficient modeling in 3 different scenarios: 1) modeling from scratch, 2) copying, pasting, and adapting an existing building, and 3) summarizing multiple buildings into an architectural style description, selecting a building from the style description, and adapting it.

In contrast to [KFWM17], our approach does not require rarely available point cloud scans, but uses manual instead of automatic reconstructions. Other approaches that also propose manual procedural reconstruction (e.g., [KK12], [LWW08], [DLC*15]) do not focus on efficient manual modeling. One approach to efficient manual modeling can be web-based crowdsourcing [LGMS16]. The authors, however, do not fully embrace crowdsourcing as their web application lacks editing capabilities.

Our contribution can be seen as follows:

- A web-based editor that integrates reference images, a procedural modeling language, a graphical user interface, and 3 procedural modeling modalities
- A novel algorithm that allows users to summarize multiple procedural models into an architectural style description, which can be used to model new buildings of a similar style more efficiently

2. Quick-Pro-Build

2.1. Overview

Our approach allows for quick 3D procedural reconstructions of buildings using street and satellite views as reference. We propose 3 modalities for efficient and realistic urban reconstruction. Figure 1 illustrates each modality and its requirements. In *Modality 1*, the users model buildings from scratch. *Modality 2* allows users to

DOI: 10.2312/egs.20231001

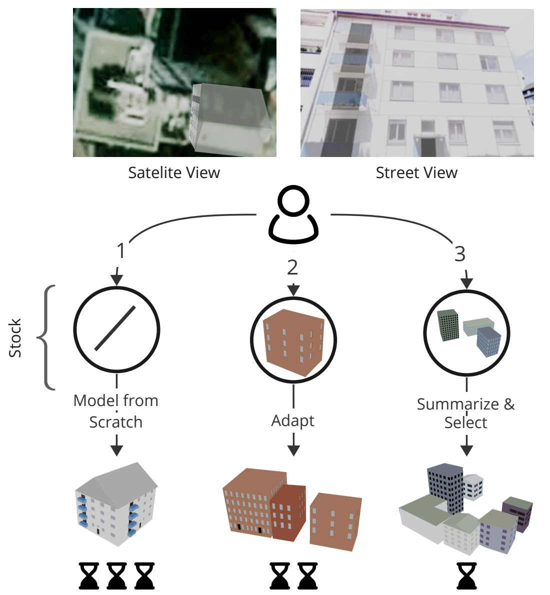
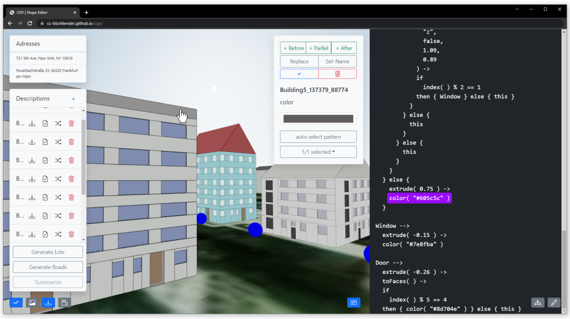


Figure 1: *Approach Overview*



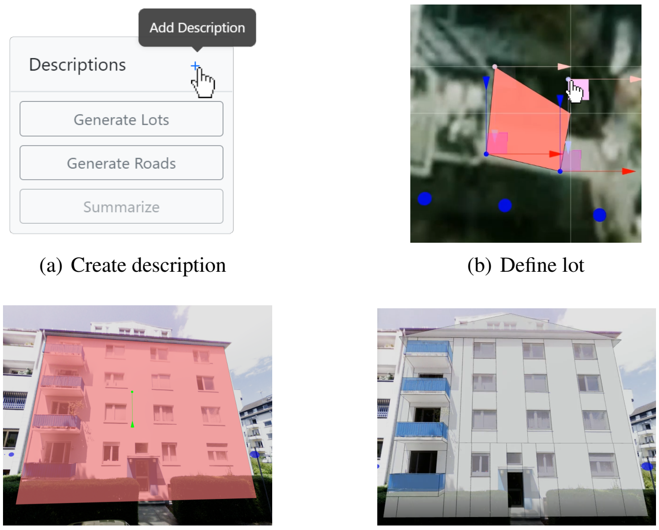
```
              "z",
              false,
              1.09,
              0.89
            ) ->
            if
              index( ) % 2 == 1
            then { Window } else { this }
          }
        } else {
          this
        }
      } else {
        this
      }
    }
  } else {
    extrude( 0.75 ) ->
    color( "#605c5c" )
  }

Window -->
  extrude( -0.15 ) ->
  color( "#7e8fba" )

Door -->
  extrude( -0.26 ) ->
  toFaces( ) ->
  if
    index( ) % 5 == 4
  then { color( "#8d704e" ) } else { this }
```

Figure 2: *Our web-based tool.*

model buildings by adapting existing models. This modality improves the efficiency of modeling, but depends on the availability of similar buildings. *Modality 3* enables users to summarize multiple existing buildings into an architectural style description. The style description is expressed by conditionals and probabilities and is used to achieve the desired reconstruction with fewer manual corrections than in *Modality 2* by learning the correct style for the new model from the preference of the user. The following sections introduce our web-based graphical interface (Section 2.2) and our 3 modeling modalities (Section 2.3).

2.2. Tool

Our tool provides a graphical user interface and a text-based interface for procedural modeling. Figure 2 illustrates our web-based tool with both interfaces and a sample scene from Frankfurt, Germany. The text-based interface on the right allows precise manipulation of the description, while the graphical user interface on the left enables a more intuitive editing experience. The graphical user interface integrates satellite and street view imagery and allows direct modification with 3D handles and the WYSIWYG concept [SCC06]. Our tool is implemented in Typescript, it uses React and Next.js for the user interface, Three.js for rendering, and React-Three-Fiber to connect the 3D and 2D interfaces. The source code is available at `https://github.com/cc-bbohlender/cgv`.



(a) Create description

(b) Define lot

(c) Façade modeling

(d) Final model

Figure 3: *Modeling in* Modality 1.

2.3. Modeling

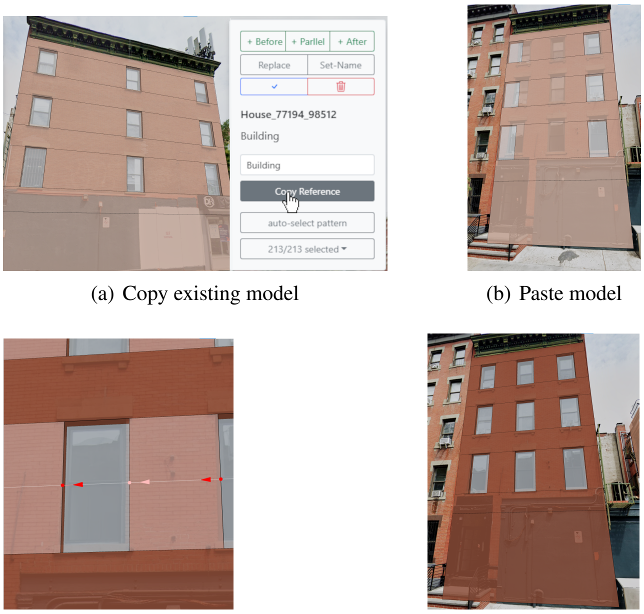
2.3.1. Modality 1 (From Scratch) and Modality 2 (Adapt)

In *Modality 1*, illustrated in Figure 3, users can model buildings from scratch. The process starts by creating a description (Figure 3(a)) and defining the lot (Figure 3(b)), using the satellite view as a reference, which allows the alignment of the model with the original building lot. Users can then switch to the street view reference to align modeling transformations, such as extrude and split (Figure 3(c)), to achieve an output (Figure 3(d)) that matches the various elements of the building.

Modality 2, illustrated in Figure 4, allows users to model a building by adapting an existing model. First, users make a copy of an existing description (Figure 4(a)) and paste it into a new description (Figure 4(b)). Users can then adapt the parameters of the transformations, particularly splits (Figure 4(c)), to finalize the model (Figure 4(d)). This modality can only be applied if a building similar to the target building exists.

2.3.2. Modality 3 (Summarize and Select)

In *Modality 3*, illustrated in Figure 5, users model buildings in three steps: Summarize, Select and Adapt. In the first step (Figure 5(a)), users select at least two models with a similar architectural style and summarize them into an architectural style description. We express this style description in our procedural language using stochasticity. In the second step (Figure 5(b)), users select one proposed instance from the style description. The preference for one instance is used to generate new instances. The users can



(a) Copy existing model (b) Paste model

(c) Adapt model (d) Final model

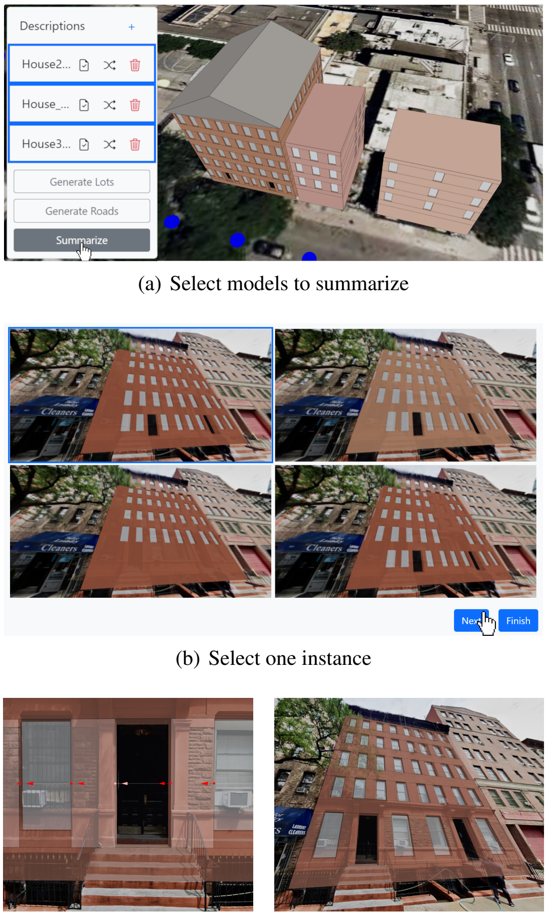
Figure 4: *Modeling in modality 2.*

repeat this step until they find the best match for the target building. In the last step, the users can make final modifications (Figure 5(c)) to achieve the desired model (Figure 5(d)).

The summarize step is enabled by a novel algorithm that computes the architectural style descriptions from multiple buildings. First, the algorithm breaks down each description into lists of sequential transformations. Next, the algorithm aligns all lists using a *difference algorithm* [Mye86], ensuring that similar transformations appear at the same position in each list. The similarity between transformations is computed based on their type and parameters. The resulting aligned lists can be represented as a table where the columns only contain similar transformations. Row cells are left empty if no matching transformation is found for one row. Next, the algorithm computes the similarities between all transformations in each row. Unsimilar rows are split horizontally into two subgroups. Similar rows are split vertically around the two most similar transformations into three subgroups. This table is split recursively until each group only contains similar transformations, and the recursive splits create a tree structure. Finally, the algorithm re-translates the tree into a description. Nodes with more than one outgoing branch are combined using a stochastic switch, expressed by, for instance:

$$\{\ 50\% : color(\text{"\#0F0"})\ \ 50\% : extrude(2)\ \}$$

In the select step, the user can retrieve a concrete description for a specific building from the style description. The stochastic elements of the style description can be represented as a decision tree. One path in the tree represents one building. We apply a learning-based algorithm [For96] to find a model the user is satisfied with, by having the user repeatedly choose one of multiple generated buildings.



(a) Select models to summarize

(b) Select one instance

(c) Adapt model (d) Final model

Figure 5: *Modeling in modality 3.*

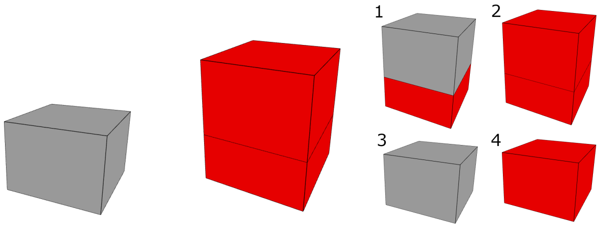
Example. In this example, a user summarizes two buildings. A gray 6-meters high building described by

$$Building1 \longrightarrow color(\text{"\#AAA"}) \rightarrow extrude(6)$$

and a 6-meters high building with a 4-meters deep basement described by

$$Building2 \longrightarrow extrude(6) \mid extrude(-4)$$

*Building*1 is illustrated in Figure 6(a) and *Building*2 in Figure 6(b). In all cases, red is the default color. We rewrite both de-



(a) *Building*1 (b) *Building*2 (c) Style variations

Figure 6: *Example input and output*

scriptions as linear lists of transformations and apply the *difference algorithm* on the 3 lists to find their best possible alignments. The aligned lists are then recursively grouped based on the similarity of the transformations.

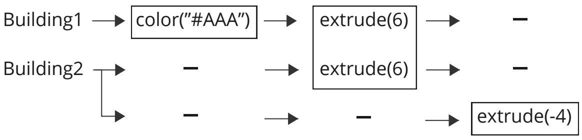


Figure 7: *Example grouped transformations.*

Figure 7 illustrates the transformations and corresponding groups indicated using boxes. For instance, the $extrude(6)$ transformations were aligned and grouped due to matching transformation type and parameter. In contrast, the $color(\text{”\#AAA”})$ and $extrude(-4)$ transformations were not grouped with any other transformation. Re-translating the grouped transformations results in

$$Building \longrightarrow \{\ 50\% : color(\text{”\#AAA”})\ 50\% : noop\ \} \rightarrow extrude(6) \mid \{\ 50\% : extrude(-4)\ \}$$

The style descriptions starts with a stochastic switch combining $color(\text{”\#AAA”})$ and *noop* (no-operation) both with 50% probability, followed by the merged $extrude(6)$ transformation. Finally, a second stochastic switch contains $extrude(-4)$ and nothing both with 50% probability. Figure 6(c) illustrates all possible instances that are derivable from this style description. For instance, following the second option (noop) for the first stochastic switch and the first option for the second stochastic switch results in

$$Building \longrightarrow extrude(6) \mid extrude(-4)$$

This is illustrated in Figure 6(c2). The path through the stochastic switches taken for the model in Figure 6(c2) can be encoded as $[1,0]$, where 1 encodes the second option for the first switch and 0 encodes the first option for the second switch. This path representation is used as the genetic code for a building inside the genetic algorithm. The algorithm finds the preferred building by randomly generating a population and procreating the user-selected instance while keeping a slight probability of variation until users are satisfied with the model.

3. Conclusion

In this work, we have proposed a web-based approach for quick procedural 3D reconstruction of buildings. Our approach combines openly available urban imagery, a web-based graphical interface, and 3 modalities for interactive procedural modeling with a novel summarization and selection algorithm.

We conducted a preliminary evaluation of our tool with 10 participants from different backgrounds and an average age of 30.5 years. Our evaluation shows promising results with noticeable efficiency improvements going from *Modality 1* to *Modality 2* to *Modality 3*. On average, participants reduced their modeling time in Modality 2 by 76% percent. In Modality 3, participants could reduce modeling time by another 9% even though they only summarized the two buildings modeled in Modality 1 and 2. We believe that the quality of models achievable with Quick-Pro-Build suffices for many use cases, while the efficiency improvements enable these use cases at scale. Participants envisioned use cases like “Delivery Robots”, “Urban Planning”, “Fire Departments”, “Crisis Planning”, and “Advanced Street View”.

Acknowledgements

This work has been funded by the LOEWE initiative (Hesse, Germany) within the emergenCITY centre.

References

[ARB07] Aliaga D. G., Rosen P. A., Bekins D. R.: Style grammars for interactive visualization of architecture. *IEEE transactions on visualization and computer graphics 13*, 4 (2007), 786–797. 1

[DLC*15] Dang M., Lienhard S., Ceylan D., Neubert B., Wonka P., Pauly M.: Interactive design of probability density functions for shape grammars. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 1–13. 1

[For96] Forrest S.: Genetic algorithms. *ACM Computing Surveys (CSUR) 28*, 1 (1996), 77–80. 3

[KFWM17] Kelly T., Femiani J., Wonka P., Mitra N. J.: Bigsur: large-scale structured urban reconstruction. *ACM Transactions on Graphics 36*, 6 (2017). 1

[KK12] Krecklau L., Kobbelt L.: Interactive modeling by procedural high-level primitives. *Computers & Graphics 36*, 5 (2012), 376–386. 1

[LGMS16] Lu Z., Guerrero P., Mitra N. J., Steed A.: Open3d: crowd-sourced distributed curation of city models. In *Proceedings of the 21st International Conference on Web3D Technology* (2016), pp. 87–94. 1

[LWW08] Lipp M., Wonka P., Wimmer M.: Interactive visual editing of grammars for procedural architecture. In *ACM SIGGRAPH 2008 papers*. 2008, pp. 1–10. 1

[MWH*06] Müller P., Wonka P., Haegler S., Ulmer A., Van Gool L.: Procedural modeling of buildings. In *ACM SIGGRAPH 2006 Papers*. 2006, pp. 614–623. 1

[Mye86] Myers E. W.: An o(nd) difference algorithm and its variations. *Algorithmica 1* (1986), 251–266. 3

[NDBP15] Nguyen Q.-D., Devaux A., Bredif M., Paparoditis N.: 3d heterogeneous interactive web mapping application. In *2015 IEEE Virtual Reality (VR)* (2015), IEEE, pp. 323–324. 1

[SCC06] Simonyi C., Christerson M., Clifford S.: Intentional software. In *Proceedings of the 21st annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications* (2006), pp. 451–464. 2

[SSH17] Sayyad E., Sen P., Höllerer T.: Panotrace: interactive 3d modeling of surround-view panoramic images in virtual reality. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology* (2017), pp. 1–10. 1

[YC16] Yuan J., Cheriyadat A. M.: Combining maps and street level images for building height and facade estimation. In *Proceedings of the 2nd ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics* (2016), pp. 1–8. 1

[YNK*18] Yao Z., Nagel C., Kunde F., Hudra G., Willkomm P., Donaubauer A., Adolphi T., Kolbe T. H.: 3dcitydb-a 3d geo-database solution for the management, analysis, and visualization of semantic 3d city models based on citygml. *Open Geospatial Data, Software and Standards 3*, 1 (2018), 1–26. 1