

University of Redlands

InSPIRe @ Redlands

---

MS GIS Program Major Individual Projects

Theses, Dissertations, and Honors Projects

---

8-2019

## Visualizations of Downtown San Bernardino and a Proposed Development Using CityEngine

Max Christian Babcock  
*University of Redlands*

Follow this and additional works at: [https://inspire.redlands.edu/gis\\_gradproj](https://inspire.redlands.edu/gis_gradproj)



Part of the [Geographic Information Sciences Commons](#)

---

### Recommended Citation

Babcock, M. C. (2019). *Visualizations of Downtown San Bernardino and a Proposed Development Using CityEngine* (Master's thesis, University of Redlands). Retrieved from [https://inspire.redlands.edu/gis\\_gradproj/288](https://inspire.redlands.edu/gis_gradproj/288)



This work is licensed under a [Creative Commons Attribution 4.0 License](#).

This material may be protected by copyright law (Title 17 U.S. Code).

This Thesis is brought to you for free and open access by the Theses, Dissertations, and Honors Projects at InSPIRe @ Redlands. It has been accepted for inclusion in MS GIS Program Major Individual Projects by an authorized administrator of InSPIRe @ Redlands. For more information, please contact [inspire@redlands.edu](mailto:inspire@redlands.edu).

University of Redlands

**Visualizations of Downtown San Bernardino  
and a Proposed Development Using CityEngine**

A Major Individual Project submitted in partial satisfaction of the requirements  
for the degree of Master of Science in Geographic Information Systems

by

Max Christian Babcock

Mark Kumler, Ph.D., Committee Chair

Douglas Flewelling, Ph.D.

August 2019

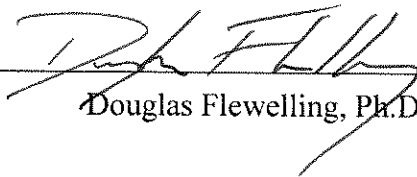
**Visualizations of Downtown San Bernardino  
and a Proposed Development Using CityEngine**

Copyright © 2019

by

Max Christian Babcock

The report of Max Christian Babcock is approved.



---

Douglas Flewelling, Ph.D.



---

Mark Kumler, Ph.D., Committee Chair

August 2019



## **Acknowledgements**

Thank you, Andrea Alvarado, for helping with all the logistics paperwork and encouragement throughout this program. Thank you to Mark Kumler for being my advisor and helping me create this project. Thank you also to Keera Moorish for helping me debugging and solving numerous CityEngine challenges and problems. I would also like to thank Nate Strout for helping with ArcGIS Pro challenges. Lastly, I would like to thank Douglas Flewelling for being my second reader and helping throughout this thesis project process with your humor and cribbage games.

My Cohorts members; Jonah, Anthony, Alex, Cristhian, Will, Mark, Abigail, and Zemen thank you all for helping me during this journey. I could not have finished without you all.

This project is dedicated to my parents, Carin and Chris Babcock for their love and support throughout my master's candidacy.



# **Abstract**

## **Visualizations of Downtown San Bernardino and a Proposed Development Using CityEngine**

by

Max Christian Babcock

Cities are experiencing increasing growth in population and business infrastructure. These changes have profound impacts on urban planners and stakeholders alike, in how they view and conceptualize potential new developments. In the past, the downtown area of the City of San Bernardino would take on new projects only having a rendering of the proposed building(s), making it time consuming and difficult to understand the wider impact on the surrounding areas. Without view analyses these developments could potentially result in termination due to deadlines or loss of interest from stakeholder. This project addressed this issue by creating 3D renderings of the area using CityEngine and performing various visual analyses for new development(s). Having CityEngine will deduct meeting time and effectively answer visual questions their various stakeholders have in regard to the developments or cityscape of downtown San Bernardino area. These conclusions of these findings were significant to the downtown City of San Bernardino, and the project was able to be created with the data provided. The data also allowed the project to and create the cityscape of the downtown area and to preform various visual analyses to solidify the project's fruition.





# Table of Contents

<b>Chapter 1 – Introduction .....</b>	<b>1</b>
1.1 Client.....	1
1.2 Problem Statement.....	1
1.3 Proposed Solution .....	1
1.3.1 Goals and Objectives .....	2
1.3.2 Methods.....	2
1.4 Audience .....	3
1.5 Overview of the Rest of this Report .....	3
<b>Chapter 2 – Background and Literature Review .....</b>	<b>5</b>
2.1 San Bernardino.....	5
2.2 Three Dimensional Visualization .....	7
2.3 Intervisibility and Usefulness of 3D in City Models .....	8
2.4 Summary .....	11
<b>Chapter 3 – Systems Analysis and Design.....</b>	<b>13</b>
3.1 Problem Statement.....	13
3.2 Requirements Analysis .....	13
3.3 System Design .....	14
3.4 Project Plan .....	15
3.5 Summary .....	16
<b>Chapter 4 – Database Design.....</b>	<b>17</b>
4.1 Conceptual Data Model .....	17
4.2 Logical Data Model .....	17
4.3 Data Sources .....	18
4.4 Data Collection Methods .....	19
4.5 Data Scrubbing and Loading .....	19
4.6 Summary .....	19
<b>Chapter 5 – Implementation.....</b>	<b>21</b>
5.1 ArcGIS Pro.....	21
5.2 CityEngine Scene.....	23
5.3 Streets Layer .....	24
5.4 Building Layer .....	30
5.5 New Development .....	36
5.6 Summary .....	37
<b>Chapter 6 – Results and Analysis.....</b>	<b>38</b>
6.1 CityEngine Scene.....	38
6.2 Development .....	39
6.3 View Analyses .....	41
6.4 Summary .....	46
<b>Chapter 7 – Conclusions and Future Work.....</b>	<b>47</b>
7.1 Conclusion .....	47
7.2 Future Work .....	47

<b>Works Cited .....</b>	<b>49</b>
<b>Appendix A. Footprint with Height Workflow .....</b>	<b>50</b>
<b>Appendix B. Coloring Parcels using CGA code.....</b>	<b>51</b>
<b>Appendix C. Building footprint code .....</b>	<b>52</b>
<b>Appendix D. Building footprint code from Esri.lib .....</b>	<b>54</b>

## Table of Figures

Figure 1-1: Study Area of Downtown San Bernardino Streets Clockwise 9 <sup>th</sup> street, Waterman avenue, Rialto Ave, and the 215 .....	2
Figure 2-1: City boundary's (Top) and Extent of Downtown San Bernardino for the Project (Bottom).....	6
Figure 2-2: A 3D representation of the Abu Dhabi development (Kuntze et al., 2012, p. 292). .....	8
Figure 2-3: A Promotional Site for City of Tokyo (Shiode, 2000, p.267).....	9
Figure 2-4: 3D Marketing Tool of Potential Areas of Development in Japan (Shiode, 2000, p.267) .....	9
Figure 2-5: Intervisibility effect on the targeted building (in Red) then the dark blue is visible and light blue is invisible. The table describes what points are affected and are not affected by the targeted building. (Wassim, 2011, p.9) .....	11
Figure 4-1: Conceptual Model .....	17
Figure 4-2: Logical Model .....	18
Figure 5-1: Clipping layers .....	21
Figure 5-2: Building a LAS dataset .....	22
Figure 5-3: Building a DEM dataset .....	22
Figure 5-4: Feature Class to Shapefile Tool .....	23
Figure 5-5: Creating a Scene.....	24
Figure 5-6: Cleanup Graph Tool.....	24
Figure 5-7: Resolving Conflicting Shapes - Incorrect .....	25
Figure 5-8: Resolving Conflicting Shapes - Correct .....	26
Figure 5-9: Merge Node - Before .....	26
Figure 5-10: Merge Node - After.....	27
Figure 5-11: Block Creation Activate.....	28
Figure 5-12: Block Deletion Disabled .....	28
Figure 5-13: Smart Road Ending .....	29
Figure 5-14: Crossing Road Ending .....	29
Figure 5-15: Colorized Parcels .....	30
Figure 5-16: Height Difference Real Height and Height Given by The Attribute ....	31
Figure 5-17: Facades Applied and Correct Building Height .....	32
Figure 5-18: Tilted Building .....	33
Figure 5-19: Corrected Tilt .....	33
Figure 5-20: Missing Building.....	34
Figure 5-21: Created Building .....	35
Figure 5-22: Carousel Mall Before .....	36
Figure 5-23: Carousel Mall After .....	36
Figure 6-1: CityEngine Scene Overview .....	38
Figure 6-2: CityEngine Scene Detailed View .....	39
Figure 6-3: Carousel Mall - Before.....	40
Figure 6-4: Carousel Mall - After .....	40
Figure 6-5: Carousel Mall Before Detailed .....	41
Figure 6-6: Carousel Mall After Detailed.....	41

Figure 6-7: Viewshed Creation Low Angle ..... 42

Figure 6-8: Viewshed Creation High Angle..... 43

Figure 6-9: View Dome Creation..... 43

Figure 6-10: View Corridor..... 44

Figure 6-11: Morning Shadow ..... 45

Figure 6-12: Afternoon Shadow ..... 46

**List of Tables**

Table 1. Deliverables for the Client ..... 14

Table 2. Project Timeline..... 16

Table 3. Data Description ..... 18

Table 4. Unclean vs. Clean Data..... 19



## List of Acronyms and Definitions

2D	Two-dimensional
3D	Three-dimensional
ADA	American Disability Association
AR	Augmented Reality
CAD	Computer Aided Design
CGA	Computer Generated Architecture
.cpg	Code Page file
.dbf	Database file
DEM	Digital Elevation Model
DSM	Digital Surface Model
LAS	Lidar Data Exchange file
.las	Lidar file format
.lib	Library file
.img	Image file
.pri	Package Resource file
.sbn	Binary Data file
.sbx	Esri Spatial Index file
.shp	Shapefile
.shp.xml	Shapefile Extensible Markup Language
.shx	Shapefile Index
.tiff	Tagged Image format file
VR	Virtual Reality





# **Chapter 1 – Introduction**

Local and national governments have been using two-dimensional (2D) visualization for their physical assets and properties. Recently, technological advancements have allowed governments to view their assets and properties in three-dimensions (3D) using Esri's CityEngine software. CityEngine allows governments to generate models of buildings and city assets to create their city accurately. CityEngine also allows governments to plan and implement future developments that helps them to conduct analysis and receive constructive feedback from stakeholders and people of the area. Governments are also using CityEngine to create a master plan of the city with attribute rich data attached to each building for performing analyses. CityEngine can procedurally generate assets for potential development sites with Computer Generated Architecture (CGA); this allows newly created features to be modified and then instantly applied across a mass amount of assets.

## **1.1 Client**

The client and the point of contact for this project was Mitch Cochran, the Director of Information Technology with the City of San Bernardino, California. The city's planning department was having problems visually understanding the impact of new development. Cochran was responsible for providing data and access to other city departments and private utility companies for the select records of their assets in the city's boundaries. He will become the chief administrator of the CityEngine database and files once the project is delivered to the city.

## **1.2 Problem Statement**

Currently, the City of San Bernardino had employed a traditional method of choosing their models by looking at freestanding renderings of potential developments with no context to already existing buildings around a proposed area, thus, making it more difficult for the city to conclude on the verdict of projects in a timely matter. Delay this let job-generating projects go by the wayside and slowed the advancement and prosperity of the City. The City of San Bernardino was in need of a tool that could visually illustrate 3D modeling procedural techniques and could processes and can create a more accurate and realistic model of the City of San Bernardino (Esri.com, 2018).

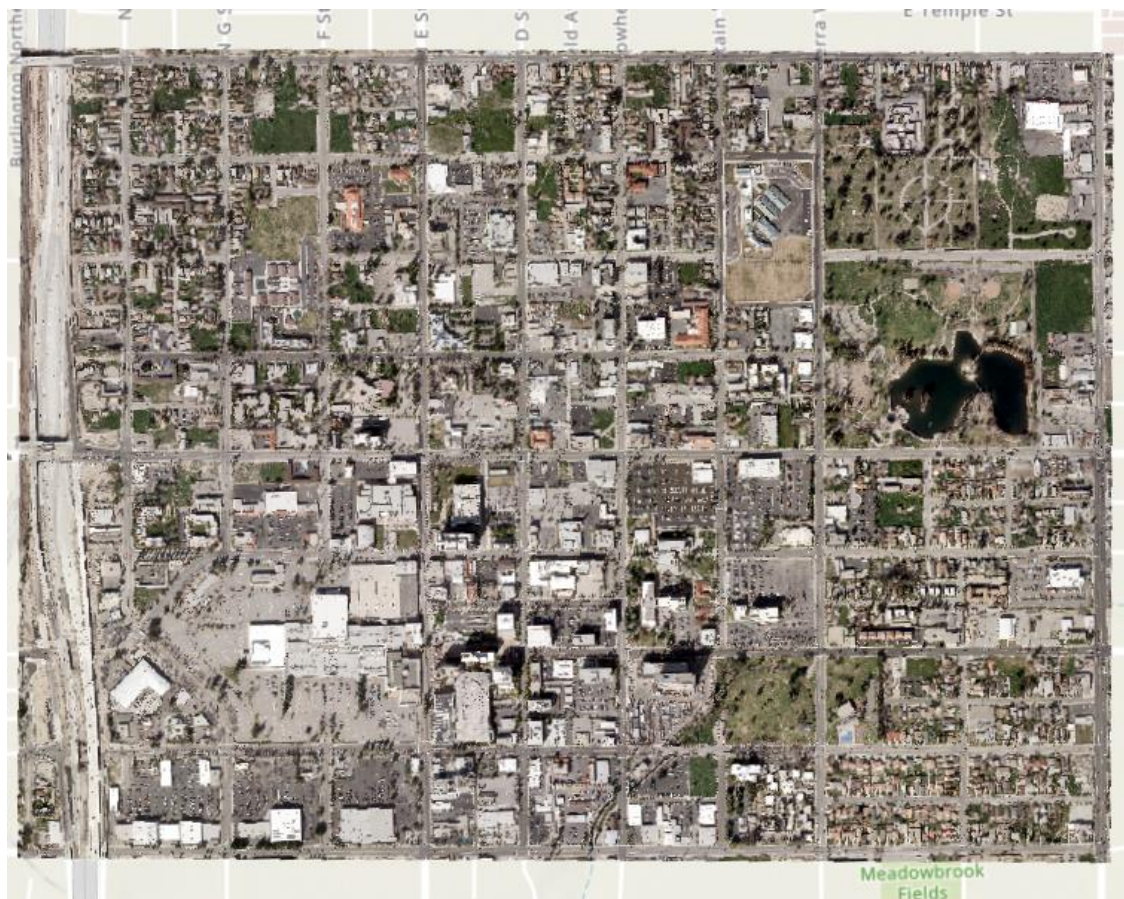
## **1.3 Proposed Solution**

The solution was to design a CityEngine model of San Bernardino, which would help speed up and improve decision making. Creating this 3D model in CityEngine helped recreate the city in various ways to illustrate why development or updating structures was needed, CityEngine's 3D modeling also showed how that change could increase the prosperity of the City of San Bernardino. A 3D system has the potential to create various analyses on structures and projects like shadow projection, American Disability Association (ADA) compatibility, green space potential, solar panel placement, cadastral,

and many other analyses. Mitch Cochran's expectations were for a system to visually and functionally showcase the city and what it could become in the future.

### 1.3.1 Goals and Objectives

The primary purpose of this project was to create an accurate 3D version of downtown San Bernardino using CityEngine and ArcGIS Pro. The secondary purpose of this project was to create a manual and a reference sheet of how to create and categorize buildings and defined features throughout the digital rendering of downtown San Bernardino. To demonstrate the utility of the model, visual analyses was performed on a new development that would replace an existing building in the downtown area. These various visual analyses are necessary to complete and solidify the effectiveness of the model.



**Figure 1-1: Study Area of Downtown San Bernardino Streets Clockwise 9<sup>th</sup> street, Waterman avenue, Rialto Ave, and the 215**

### 1.3.2 Methods

The most effective methodology for this project was be the rapid prototype method. The method was to create prototype the model tests the models features to make sure that each component is correctly edited and visually correct. Then upon correction analyses

will be performed on the data to show change. Any refinements will be completed as well at this time. Then with the edits and analyses completed a deliverable product will be sent to the client.

## **1.4 Audience**

The primary audience for this report is Mitch Cochran and San Bernardino City personnel. The secondary audience was county and city planners, consultant planning firms, government agencies, and architecture firms. Other potential audience members are those interested in understanding CityEngine and 3D applications to an environment for analysis usage in their environment.

## **1.5 Overview of the Rest of this Report**

Chapter 2 provides background information on how to create a CityEngine scene from scratch and essential information vital to this subject matter. Chapter 3, the project plan, discusses the system requirements for creating the model to perform the analysis and the system design for the project. Chapter 4 talks about database design, conceptual models, and data sources, and chapter 5 talks about implementation. Chapter 6 discusses the results and analysis. Chapter 7 provides the conclusion and information about future work.



## **Chapter 2 – Background and Literature Review**

The objective of this project is creating a 3D environment that can visually illustrate the story of an area with little use of 2D. City planners and developers in San Bernardino have mostly used 2D visualizations.

### **2.1 San Bernardino**

The City of San Bernardino is located in the San Bernardino Valley 60 miles east of Los Angeles, California, at an elevation of 1,049 feet above sea level and with an area of approximately 62.24 square miles (City of San Bernardino, 2019). For this project, the location is focused on the downtown area bounded by Interstate 215 on the west, 9<sup>th</sup> street on the north, Waterman Avenue on the east, and Rialto Avenue on the south. (Figure 2-1).





The labor force in San Bernardino is, “84,400 with 77,500 employed, resulting in an 8.2 percent unemployment rate. For the past five years, the city has seen a steady 2 percent annual decrease in unemployment” (City of San Bernardino, 2019). The City of San Bernardino has a median household income of \$41,027 as of the 2017 census (U.S. Census Bureau, 2018). Given these numbers, growth in the job market will lead to more development and job creation with the falling unemployment numbers, and in turn, the median income will rise over time. This growth has been positive for the city and their statistics since the city’s bankruptcy; “The City initially filed for bankruptcy protection in August 2012 due to the depressed of economy, a multi-million-dollar deficit, and the loss of redevelopment funding. A focus on rebuilding public safety, economic development, and improving the quality of life for stakeholders is at the center of the plan to recovery (City of San Bernardino, 2019). With a new vision of their future, the city has been strained in the planning process. The client has indicated a need for a software solution to expedite new development proposal.

## **2.2 Three-Dimensional Visualization**

For the last decade, 3D GIS functionality has been on the market. The role of geo-information in all kinds of business processes is becoming more transparent. Such terms as “location-specific information” and “location-based services” are becoming more a part of the business language to signify the link between the virtual world of information and reality (Zlatanova *et al.*, 2002, p.1). Organizations and governments have been seeing an increase in the value of having their assets and potential new developments created in 3-dimensional software and having new questions answered with this software. For instance, Stoter and Zlatanova (2009, p.2) describe that “the need for 3D information is rapidly increasing. 2D GIS analysis has shown its limitations in some situations, e.g., noise prediction models (noise spreads out in three dimensions) (Kluijver and Stoter, 2003), water flood models, air pollution models, geological models (Van Wees, 2002), real estate market (CEUS, 2003; Stoter and Ploeger, 2003).”

The advantages of 3D maps are that they are more visually appealing to stakeholders in various situations. For displaying a development to the public, having easy access to customizable angles for viewing sides of a building and having access to different viewpoint positions can visually and effectively tell a story to people of all ages and economic backgrounds. Rather than having to create multiple 2D maps with lengthy text providing details of what is to come in the physical space. Having this lengthy text attached to the 2D map could hinder individuals or groups of people who cannot read the language or do not have the required reading level. Another advantage of 3D is the ability to display photo-realistic information about the landscape and buildings in the surrounding area. This could help in immersing a person into the environment through a picture or through virtual reality (VR) where a person can have a 360-degree view of the surroundings and be able to walk around and see the landscape and models generated in real-time. Lastly, an advantage of 3D is that a person can more easily identify topography and the elevation in the surrounding.

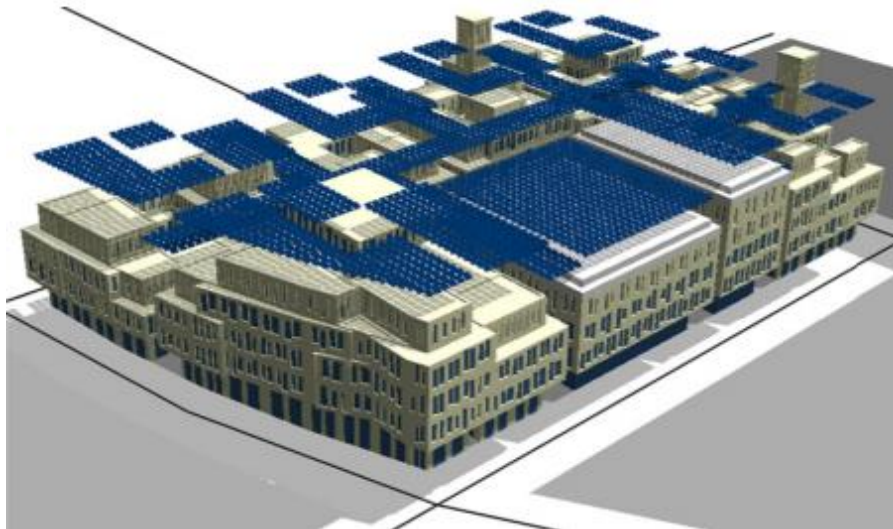
In contrast, the disadvantages of using a 3D map are that they can be challenging to see both the front and back within a single scene. Another disadvantage is that significant topographic features can obscure information and views of the background that may hold valuable information. 3D software training is more specialized and



expensive, and the cost of the software is more than \$5,000. 3D being as it is, makes a 2D map both cheaper and able to use lower DEM quality compared to 3D maps (Schobesberger & Patterson, 2006).

### 2.3 Intervisibility and Usefulness of 3D in City Models

A study involving visualization and decision support tools was published by Kunze et al. in 2012. They reported that metropolitan areas have experienced rapid growth in population, and that between now and 2030, 90% of population growth is expected to occur in cities worldwide. With this growth in cities, there will be significant impacts on the existing buildings and environment inside and surrounding the city, and urban planners must prepare. In 2007 a conference was held that focused on the visualization of future cities in which, case study was put forth at the conference to observe the process of creating a potential sustainable development in Abu Dhabi. A 3D rendering of a development with parcel lines elevated above the rooftops (Figure 2-2) was used to communicate design measurements to participants during the meeting.



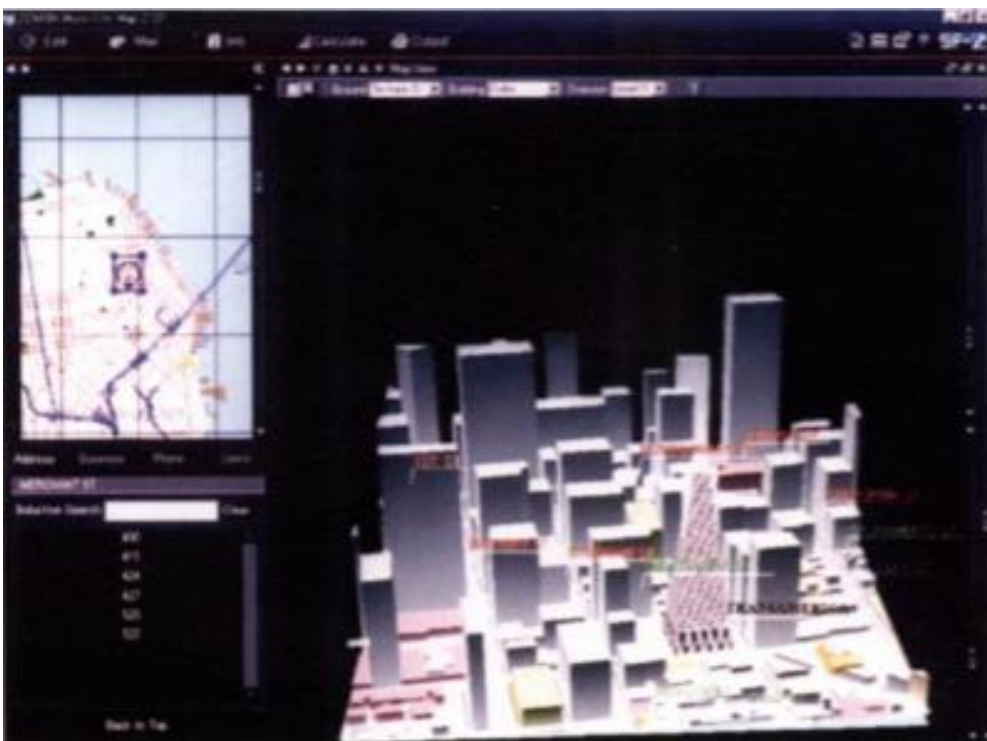
**Figure 2-2: A 3D representation of the Abu Dhabi development (Kuntze et al., 2012, p. 292).**

With the 3D rendering of the development, were and concerns were addressed immediately. The authors concluded that decision support tools are an effective means of communicating what can happen in a potential development site (Kuntze et al., 2012).

A similar study was conducted that examines cases worldwide on 3D model rendering by Shiode in 2000. There is a growing interest in urban environments for which a host of digital mapping and 3D rendering techniques are used for potential new developments. This study focuses on the effectiveness of GIS techniques, and the use of data that are utilized on 3D renderings and how they contribute to geographical analyses and planning of urban environments. 3D is gaining significant professional interest in larger cities. More than 60 large scale projects worldwide (e.g. Los Angeles, New York, and Tokyo) are using 3D applications in the four areas of commercial use, information on cities, planning use, and infrastructure services, which are shown in Figure 2-3 and 2-4.



**Figure 2-3: A Promotional Site for City of Tokyo (Shiode, 2000, p.267)**



**Figure 2-4: 3D Marketing Tool of Potential Areas of Development in Japan (Shiode, 2000, p.267)**

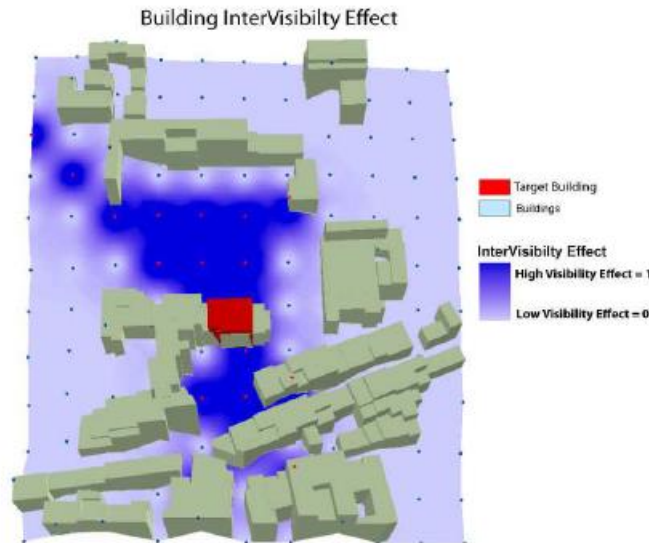
These applications are what cities are looking for in 3D renderings since these four areas (previously listed) are of greatest concern to the cities stakeholders. Having these 3D renderings reduces the number of questions about potential hazards or areas of conflict with other stakeholders. Currently, there is no clear standard in the modeling

arena. With whatever the standards are in the future, GIS technologies will doubtlessly enhance and support 3D modeling moving forward (Shiode, 2000).

In addition to Kuntz et al. and Shide, another study on the review of applications of 3D City Models by Biljecki et al. in 2015. They reported that 3D city models were not only used for visualization, but 3D was also being used for a wider range of tasks. This report investigates the state of the art of 3D utilization across various fields based on hundreds of research papers. In their research Biljecki et al. found there were advantages of 3D over 2D. Unfortunately, such research papers and information are difficult to find. Although concrete evidence of the advantages of 3D over 2D is rare, the continued use of 3D city models will provide more evidence of the advantages of 3D in the application to urban planning. (Biljecki, Stoter, Ledoux, Alatanova, & Coltekin, 2015)

In Wassim's study, visibility estimation has an impact on economic and aesthetic fields. An environment that contains a mix of artificial objects, like buildings, that are then laid on a natural ground surface, create a challenge to calculate visibility. This study presents a new method of solving this problem with vector GIS data. The vector data gives the possibility of calculating intervisibility indices as well as delineating viewshed. This new method identifies obstacles that may block the visibility of an observer situated from a vantage point in a 3D environment. This method can also calculate intervisibility on specific buildings. Using this method, a building is selected and then an intervisibility effect is performed on the building. The results of this method are in the form of a new attribute called "Affected" and is outputted to a table that the selected buildings has an affect or does not have an affect on the surrounding environment, shown in Figure 2-5.

Attributs de VisGridForBuilding									
FID	Shape	POINTID	GRID_CODE	Z	Relative_ZGH	Affected	HiddenPointat		
69	Point	8955	5472840	561,574	14,39	1	8951, 8952, 8953, 9012, 9013,		
77	Point	9013	5745791	574,8781	0,1	1	8995, 8996,		
100	Point	9134	5625875	562,8875	0,1	1	8993, 8954,		
89	Point	9074	5744918	574,5918	0,1	1	8993,		
111	Point	9192	5863129	586,4129	0,1	1	8993,		
122	Point	9249	5894113	589,5113	0,1	1	8993,		
80	Point	9018	5580632	558,1632	0,1	1	8989,		
0	Point	8636	5817961	581,8961	0,1	0			
1	Point	8637	5832262	583,3262	0,1	0			
2	Point	8638	5828184	582,8184	0,1	0			



**Figure 2-5: Intervisibility effect on the targeted building (in Red) then the dark blue is visible and light blue is invisible. The table describes what points are affected and are not affected by the targeted building. (Wassim, 2011, p.9)**

This Intervisibility method is a work in progress and the additions to this method would include a vegetation layer and building facades. These additional features would make a more accurate depiction of the intervisibility effect on the surrounding area (Wassim, 2011).

## 2.4 Summary

There is evidence that having a 3D rendering building of a city can improve the city vision of the future and make the present developments clear and easy to understand for all age groups and economic backgrounds. There are also visual analyses that can significantly benefit a new development site by making sure it does not completely block lines of the site or obstructing the line of site to a ridgeline or a skyline of another building in the immediate area or the surrounding area. From here, the next step is to create a project plan and design a system that will cater to creating a 3D city in CityEngine.



## **Chapter 3 – Systems Analysis and Design**

This chapter elaborates on the requirements and workflow that are needed to create the City of San Bernardino CityEngine project. It will also elaborate on the initial project plan and how the goals and requirements were met in the process of completing this project.

### **3.1 Problem Statement**

Currently, the City of San Bernardino had employed a traditional method of choosing their models by looking at freestanding renderings of potential developments with no context to already existing buildings around a proposed area, thus, making it more difficult for the city to conclude on the verdict of projects in a timely matter. Delay this let job-generating projects go by the wayside and slowed the advancement and prosperity of the City. The City of San Bernardino was in need of a tool that could visually illustrate 3D modeling procedural techniques and could processes and can create a more accurate and realistic model of the City of San Bernardino (Esri.com, 2018).

### **3.2 Requirements Analysis**

The objectives for the CityEngine project of the City of San Bernardino were to create a CityEngine file of their downtown area and to create a how-to workflow. The other deliverable was a demonstration of adding new buildings and an illustration of the visibility tools available in CityEngine. The requirements can be examined below in Table 1.

**Table 1. Deliverables for the Client**

Deliverable	Requirements
CityEngine File	<ul style="list-style-type: none"><li>• Accurate building locations that are level to the ground</li><li>• Accurate street data that is cleaned in CityEngine</li><li>• Accurate parcel data with colorization</li><li>• Ability to edit the file</li><li>• Extruded building layer</li><li>• Accurate terrain layer</li><li>• Generated Facades on buildings</li><li>• Generate roofs</li></ul>
“How to” Workflow	<ul style="list-style-type: none"><li>• Documented detailed workflow</li></ul>
Analysis	<ul style="list-style-type: none"><li>• Create a new structure</li><li>• Conduct viewshed analysis dome, viewshed, and view corridor</li><li>• Adding a facade</li><li>• Shadow analysis</li></ul>

For the City of San Bernardino file deliverable, they wanted to have an accurate 3D map with all the features listed in Table 1. Having a workable 3D model of the city should lead to eventual funding for an additional GIS employee to continue to create and refine the cityscape and to enhance newly created ideas for stakeholders. This would allow stakeholders to know what is to come in this physical space. The second deliverable was a how-to workflow, with clear step-by-step documentation of all the tools and edits to make a complete file. Lastly, was to performing analyses on the synthetically made buildings in the Carousel Mall area, then running both viewshed and line of sight analyses to make sure that the new building was in line with preexisting structures.

### **3.3 System Design**

There were five significant parts to this project: organizing the data provided by the city, clipping, and creating the data in ArcGIS Pro, transferring the data in .shp (Shapefile) format into CityEngine, edit the data, and then create the new development and preform

analyses on the new structure. CityEngine then created a 3D model to perform analyses on the downtown landscape. The method which was best for designing this project was the rapid prototyping method, as described by Oxford dictionary website, “a process used to build a physical model from a computer drawing by creating layers of the shape and joining them together.” This was the best method for this project since each step needed to be finished before the next, and there was no need to repeat or refine steps once they were completed. A diagram outlines the system design for the steps that were used to create this project (Figure 3-1).

### **3.4 Project Plan**

The genesis of this project was to create a proof of concept of the San Bernardino downtown area using CityEngine. This would allow city planners and GIS analysts to simulate alterations in the existing structures to visually show stakeholders the changes that a developer or a department is proposing. The project was ultimately creating a proof of concept for the City to invest in by hiring a new GIS employee and starting to develop the City and beyond. Collecting the requirements for this project included having a 3D virtual downtown file of the city with parcels, streets, buildings, and a DEM. These requirements were fulfilled through various meetings and phone calls throughout the project's creation. In the planning phase, learning CityEngine was the most critical step in furthering the project's completion. This was accomplished through independent studying and tutorial training. After learning how to use CityEngine, there was a discovery that the footprint data given did not have a height attribute for each building. Not having a height attribute on the buildings was a setback, and through time and workflows, this hurdle was overcome. In the development phase, there was much data cleaning and correction to do on streets and building. There were missing streets, streets that were not aligned correctly, and buildings that were off-kilter for the terrain, missing footprints of buildings, and building features that were misrepresented for their original design this took additional time to correct this did not alter the timeline of completion. Once the city model was edits were completed, the next step was to simulate the development of the Carousel Mall into a business park. This involved creating a new scene and removing the main structure and creating something new. This new development was quickly built with little problem and was able to showcase the potential of this property with more greenery and potential for new businesses to grow. Upon concluding this new development, various visual analyses were performed including viewshed, dome, corridor view, and shadow analyses. Project deliver when successfully in delivering on what the client wanted and they were pleased with the results but they will not be green light this project due to budget constraints and lack of resources to continue this project.



**Table 2. Project Timeline**

CityEngine Project - San Bernardino									
Month	October	November	December	January	February	March	April	May	June
Collection Requirements									
Planning									
Design									
Develop									
Deploy									

### 3.5 Summary

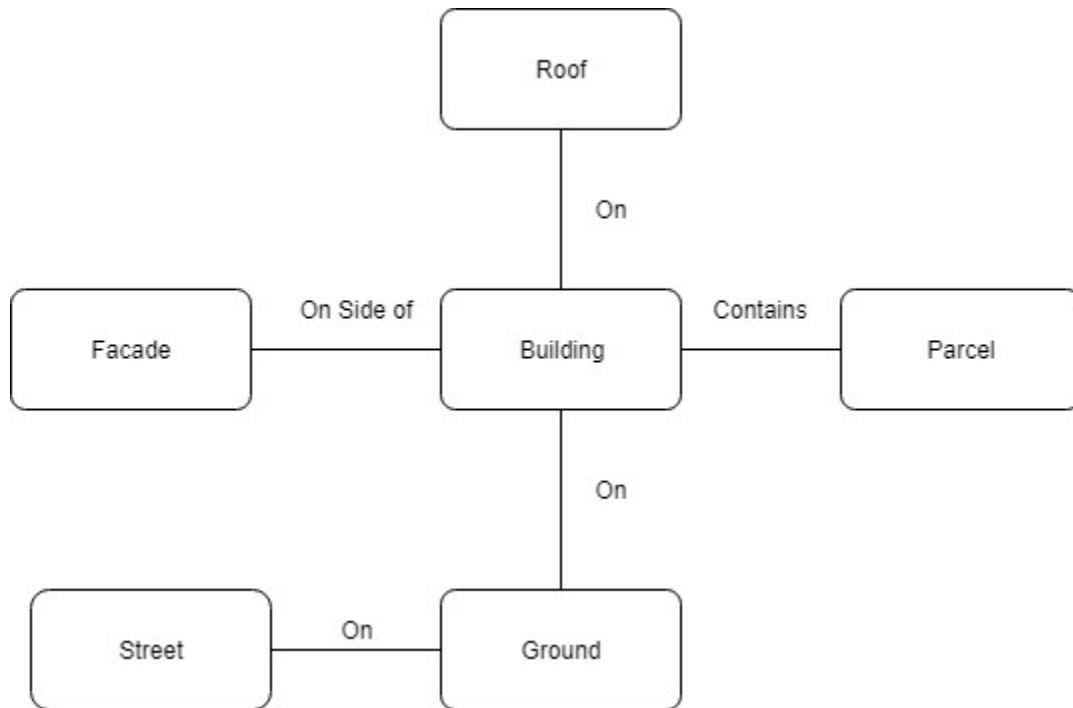
The project plan shows how this project was completed on a timeline, and each step that was needed. In the collection of requirements stage, there were outlines of precisely what the client desired in the project. In the design stage, there was time spent on creating the geodatabase that would be utilized in CityEngine. Following the design phase, the development phase was where the most considerable amount of the time was spent on the CityEngine scene this was double the amount of time than expected. The analysis portion was shorter than anticipated. Lastly, the deployment phase was where the project was fully transferred to the client, and the project was complete.

## Chapter 4 – Database Design

Preceding the work in CityEngine, a database needed to be created through ArcGIS Pro. Upon reviewing the requirements, a conceptual model was needed to explain how each component was connected. Following this, a logical model was needed to show how the data fit together and how this could become a functional database.

### 4.1 Conceptual Data Model

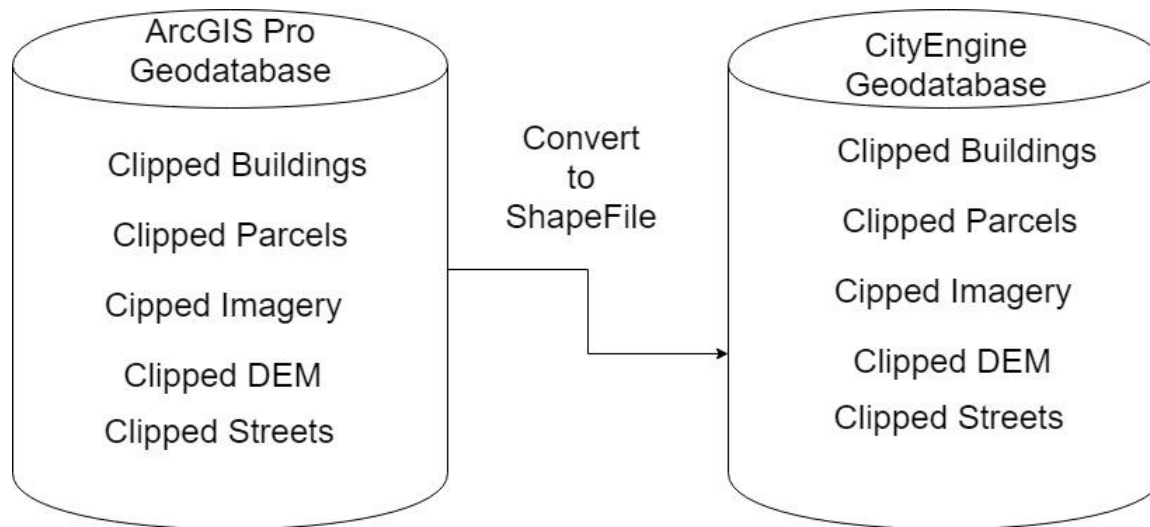
In a Cityscape there are roads that are laid on top of the ground. These roads are adjacent to buildings that are contained in parcels boundaries. On the sides of the buildings there are facades depicting the buildings details. Roofs are on top of buildings.



**Figure 3-1: Conceptual Model**

### 4.2 Logical Data Model

Each data part needed to be clipped in ArcGISPro in their respective formats (i.e., Feature Classes and Imagery). These datasets then had to be converted into .shp files where they utilized for their intended purpose in CityEngine. The main files that were being clipped were Building footprints, Parcels, DEM, Imagery, and Roads. Figure 4-2 shows that to necessary layers and imagery to successfully transfer to CityEngine.



**Figure 4-2: Logical Model**

### 4.3 Data Sources

The primary data source was from the City of San Bernardino IT department that was delivered via an external hard drive. The data that was stored on the hard drive was in three separate folders, and one contained County Lidar data that was broken into three county sections; only folder A and B were pertinent to the project. The aerial data folder contained 3cm and half foot data. Lastly, an SBdatabase contained county parcels, primary street networks, city limits, and extent clipped polygons. Footprint data was sourced on ArcGISOnline.com by the author sbreyer\_esri of Esri. In table 2 lays out the different data that was required to create the project.

**Table 3. Data Description**

Data	Description
<b>Building layer</b>	Having a z-axis
<b>Street layer</b>	Feature type Simple
<b>Imagery</b>	3cm resolution
<b>DEM</b>	3x3 feet resolution
<b>Parcel</b>	Usetype to define parcel type
<b>LAS layer</b>	30.1 million points

## 4.4 Data Collection Methods

Upon obtaining the data, there was no footprint data to be found in the files that were given. To obtain this crucial piece of data, the solution was to go to ArcGISOnline.com and search for Microsoft Building Footprints – Tiles, then open the file in ArcGIS Pro. The footprint file was then saved to the computer and was clipped to the downtown area that was initially specified.

## 4.5 Data Scrubbing and Loading

Parcel and Street data needed to be clipped to the downtown area of San Bernardino by using the clip tool in ArcGIS Pro. For the aerial data, the images needed to be mosaiced and then clipped to the downtown extent. With the data having been clipped, there still appeared missing footprint data. See table below

**Table 4. Unclean vs. Clean Data**

Unclean Data	Clean Data
Parcel Layer	Clipped Parcel
Streets Layer	Clipped Streets
Ortho. Image	Clipped Image
DEM layer	Clipped DEM
LAS layer	Clipped layer

After finding out the city did not have footprint data, the result was finding it on Esri Online by the name of sbreyer\_esri (Sean Breyer the Program Manager of the living atlas team) of Esri. This vector tile was clipped to the downtown extent and then had a workflow performed on the layer to extrude the building heights on the right coordinate system.

## 4.6 Summary

Creating and cleaning data was vital to the 3D process of creating a city scene. These processes of organizing the data and cleaning needed parts of CityEngine were necessary for optimal use in the analyses performed on such new developments to have an accurate and correct look and feel of the city and its surroundings.

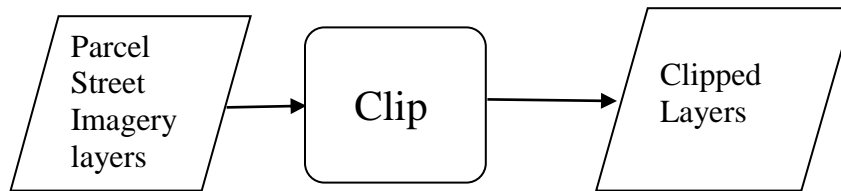


## Chapter 5 – Implementation

This style of creating 3D structures in CityEngine is not new, but the process in this project was unique and could shed light on the importance of having footprint data for governments. CityEngine also showcases the usefulness of having a 3D model of a city and how this model can improve the speed and confidence of starting a new development within the city. This chapter will focus on creating a CityEngine scene, starting with creating the data in ArcGIS Pro then converting it into to .shp files. The .shp files then create the scene in CityEngine, having buildings, streets, parcels, and terrain accurately represented in the scene. Lastly, the project creates a newly developed area, performing analyses on the newly developed area in the new scene.

### 5.1 ArcGIS Pro

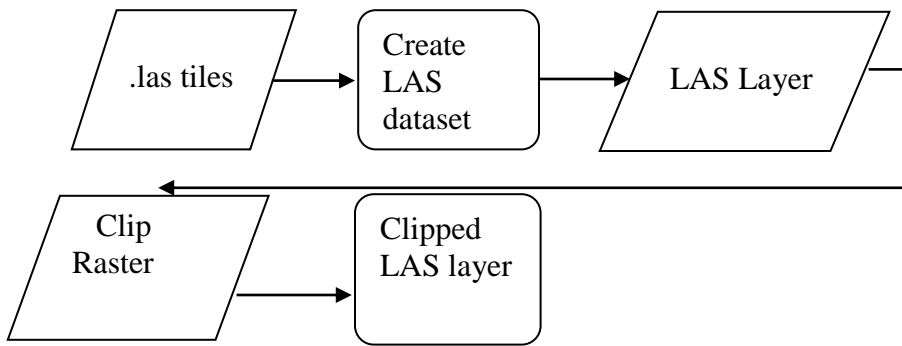
The first step was to access the data relevant to this project and input that information into ArcGIS Pro. ArcGIS Pro then clip the data to the extent of the project, and from this, the database of clipped feature classes and imagery was created. This was achieved by going to the edit tab and using the creation tool to make the extent, then this tool was applied to the streets layer, parcel layer, and imagery layer. Figure 5-1.



**Figure 5-1: Clipping layers**

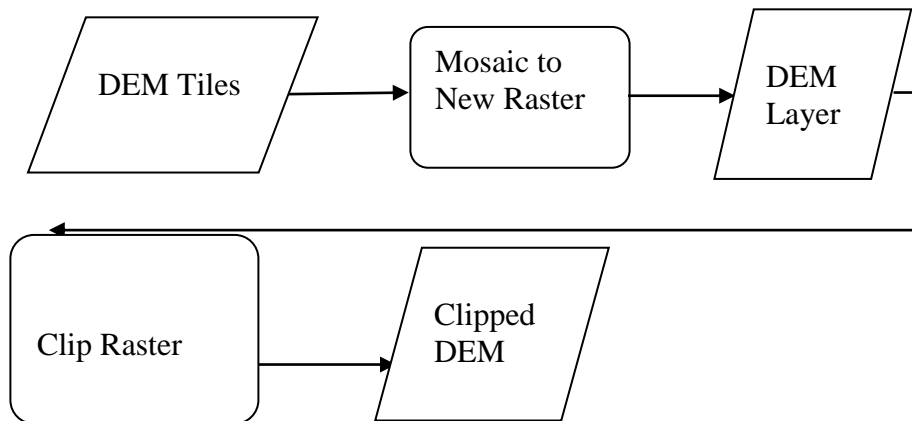
When completing the imagery file, it was essential to have the final version in a Tagged Image Format File (.tiff) for CityEngine to read when it was ready to import. This format was vital to the process because .tiff files create a graphic that CityEngine could produce for the terrain layer.

For the lidar (LAS) data, the assembly of the individual tiles needed to become one dataset. A crucial component was having the coordinate system set to NAD\_1983\_StatePlane\_California\_V\_FIPS\_0405\_Feet. This process (Figure 5-2) created a dataset for creating footprints. This data with help create the building footprints height attribute .shp file later in the implementation process.



**Figure 5-2: Building a LAS dataset**

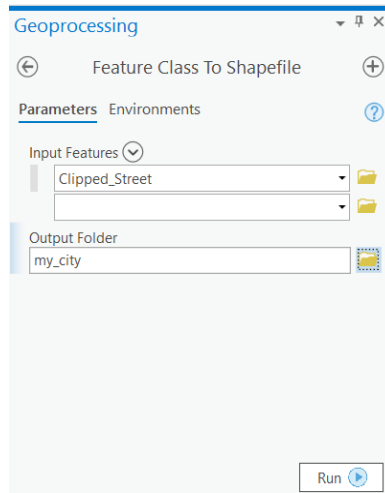
DEM data processing was similar to the LAS data workflow, but the main tool changed. The importance of creating this layer was creating the terrain to accurately represent the features of the landscape throughout the extent of the scene. See Figure 5-3.



**Figure 5-3: Building a DEM dataset**

Upon clipping the data to the desired extent, converting it to a .tiff file was necessary for the completion of the process. The main component of creating this .tiff file was having the raster size 4,000 by 4,000, which is the largest extent CityEngine can handle. If the extent is larger, the file will not render correctly. Having a smaller size less than 4,000 by 4,000 can work, but the images will be blurry and pixelated. With the completion of previous layers, the footprints were then created. The process of obtaining the footprints can be found in section 4.4 Data Collection Methods. The workflows of how to complete these steps in the process can be found in Appendix A. This process used LAS data and DEM data to find and outline footprints and to find their height using LAS.

With the parcel, street, and building layers clipped, the last step was converting these files into .shp files Using the Feature Class To Shapefile tool. Figure 5-4.

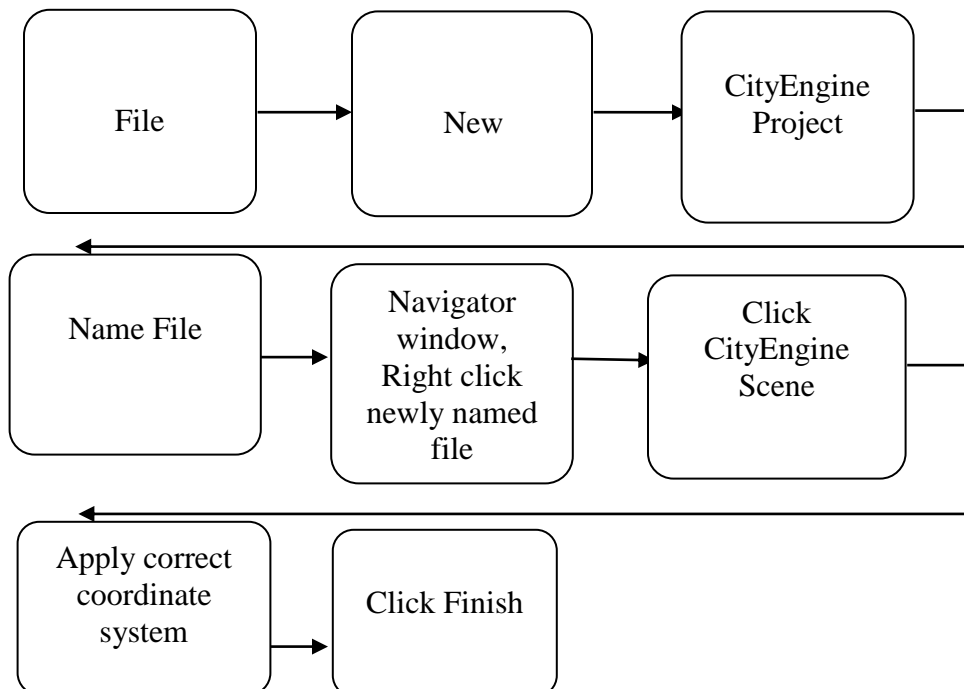


**Figure 5-4: Feature Class to Shapefile Tool**

Once the tool ran it gave the following files: Code Page File (.cpg), Database file (.dbf), Package Resource Index (.prj), Binary Data File (.sbn), ESRI Spatial Index File (.sbx), (.shp), Shapefile Extensible Markup Language (.shp.xml), and Shapefile Index File (.shx). All files were then saved to their designated folder. These files were needed to support the data files in the CityEngine scene. The data was then ready to be imported into CityEngine.

## 5.2 CityEngine Scene

With the data assembled in ArcGIS Pro, the CityEngine scene could be created. To start, the user needs to create the CityEngine file and the CityEngine scene. Figure 5-5 show the workflow creating a CityEngine Scene.



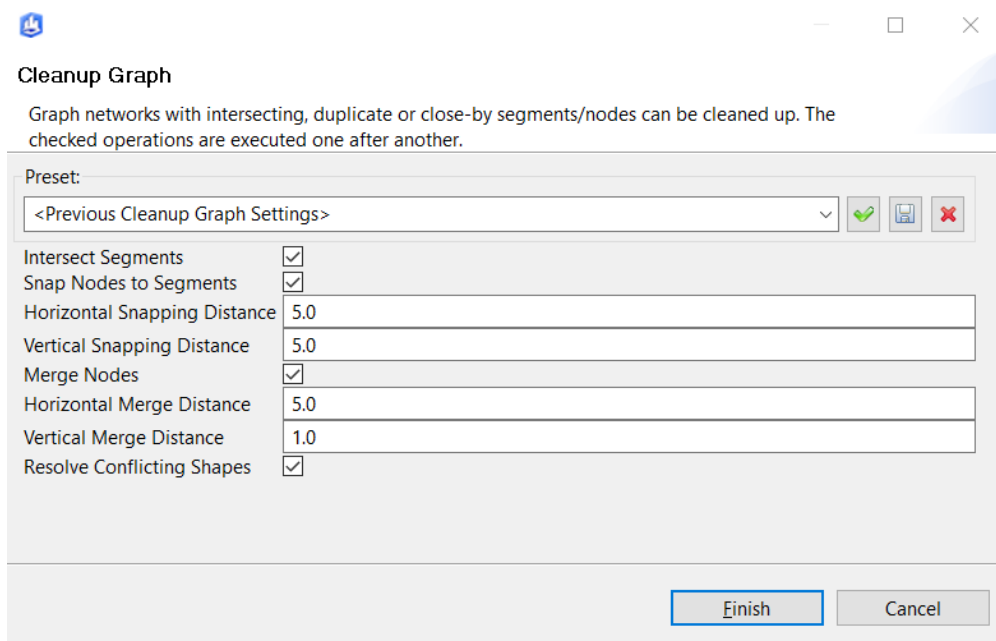


**Figure 5-5: Creating a Scene**

With having the scene created, the newly created ArcGIS Pro files were imported into the corresponding CityEngine navigator pane in their respective files. With the files in the navigator pane and having clicked on the DEM layer with the .img extension, the files were dragged and dropped it into the view window. In the pop-up window choice the texture file had to be in .tiff format.

### 5.3 Streets Layer

For the streets layer, the shapefile layer had to be dragged and dropped into the viewer. Having the streets layer in the view pane, with the terrain layer above the streets layer, the CleanUp graph tool then had to be applied to fix this issue. Once completed, the streets layer lay on top of the terrain layer. To give the streets layer a realistic feel, it was necessary to access the ESRI.lib library in the Navigator window and find the rules file; next, open the file, find the streets file and drag and drop the standard rule file onto the window scene, and then click generate. There were numerous inaccuracies, misalignments, and incorrect street segments throughout the streets layer. The CleanUp graph tool was applied to correct many of these mistakes and inaccuracies. The CleanUp graph tool also provided fixes, such as, intersect segments, snap nodes to segments, merge nodes, and resolve conflicting shapes by applying the correct rule. In the streets layer for this scene, there were no intersect segments or snap nodes to segment problems. Figures 5-6, 5-7, 5-8, 5-9, and 5-10 illustrates the CleanUp graph tool and its uses on the streets layer.



**Figure 4-6: Cleanup Graph Tool**



**Figure 5-7: Resolving Conflicting Shapes - Incorrect**



**Figure 5-8: Resolving Conflicting Shapes - Correct**

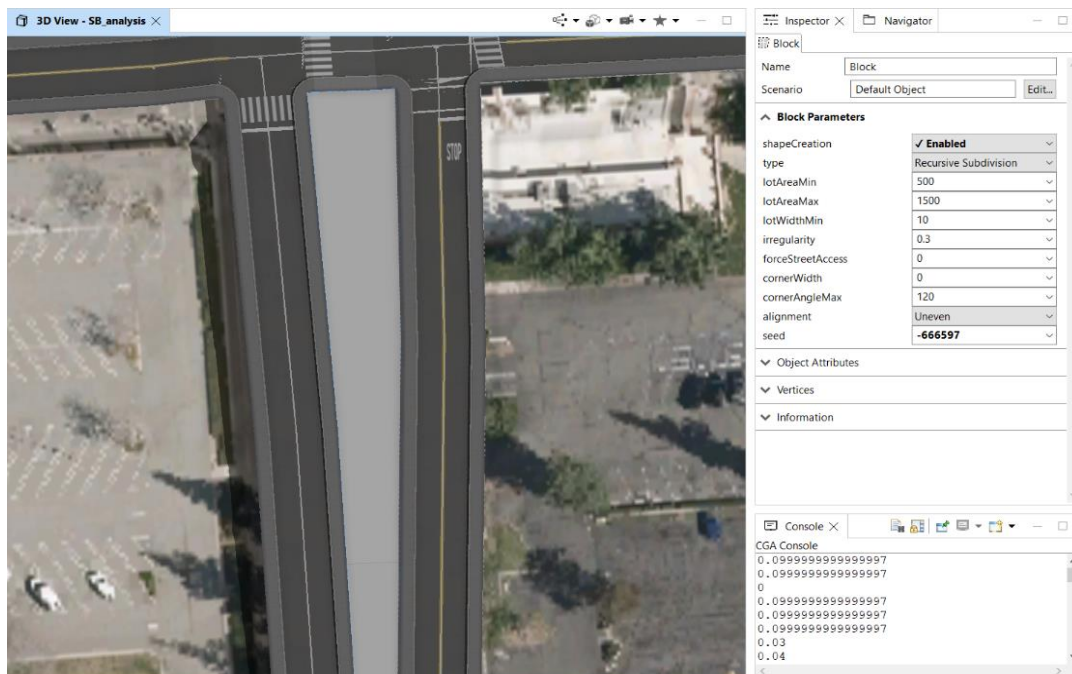


**Figure 5-9: Merge Node - Before**

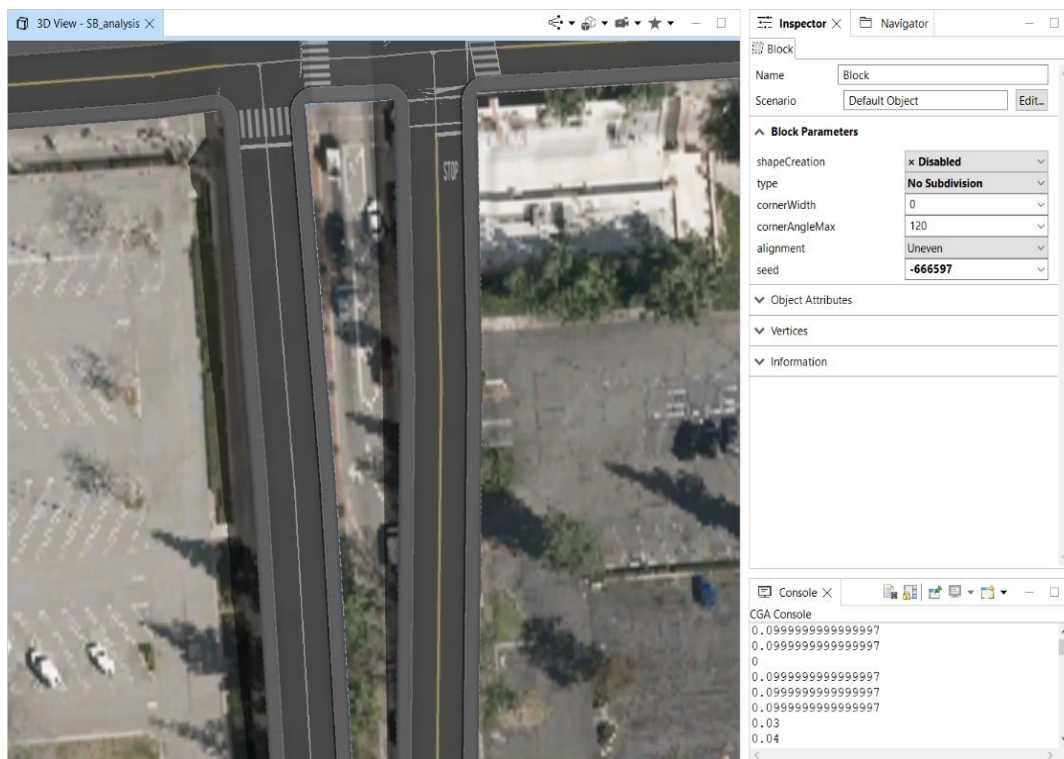


**Figure 5-10: Merge Node - After**

When editing the streets layer the layer, sometimes could create extra parcels because when moving the streets around, the streets sometimes interpreted it as a new parcel block. To fix this, the dotted line between the streets was selected then it was necessary to navigate to the Block Parameters in the inspector window, go to shapeCreation and change from active to disable, and the type needed to be changed from its current setting to no subdivision. The street was moved from off center to illustrate this problem. See Figures 5-11 and 5-12.



**Figure 5-11: Block Creation Activate**



**Figure 5-12: Block Deletion Disabled**

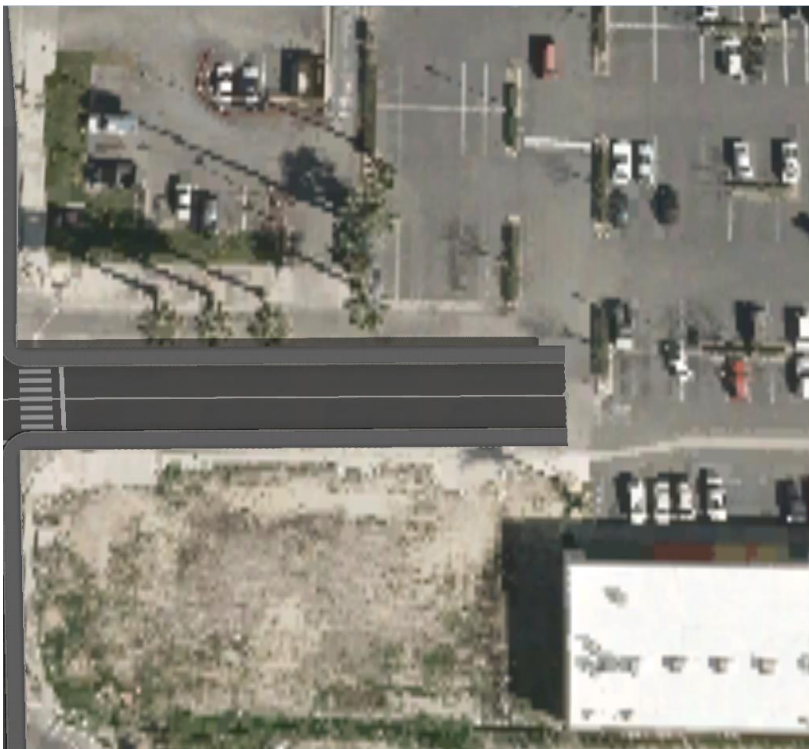
Some streets needed different types of ends, and it was possible to choose from roundabouts, crossings, or junctions as the proper end for each street. This was accomplished by going to the intersection parameters pane type from the drop-down



menu, where it was changed from “smart” to “crossing”. Figure 5-13 and 5-14 illustrate the change from smart to crossing.

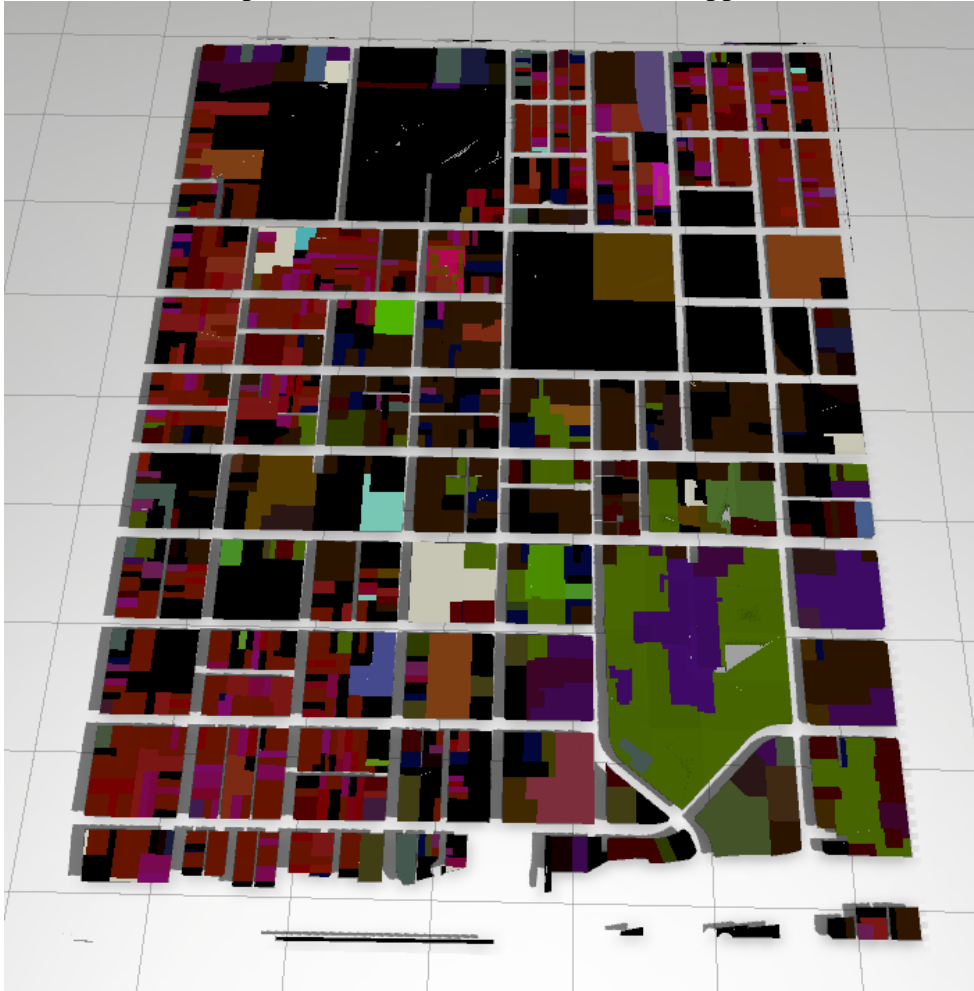


**Figure 5-13: Smart Road Ending**



**Figure 5-14: Crossing Road Ending**

Once the streets were edited, the parcel data were loaded. The parcel shapefile had to be dragged and dropped into the CityEngine scene, and once completed, the `align` shape to the terrain tool was used to finish. To color the parcels, new Computer-generated architecture (CGA) code needed to be created to colorize each type of parcel by the `TYPEUSE` attribute, which gave color and fill to each parcel Figure 5-15 shows the colored parcel. The CGA code is found in Appendix B.

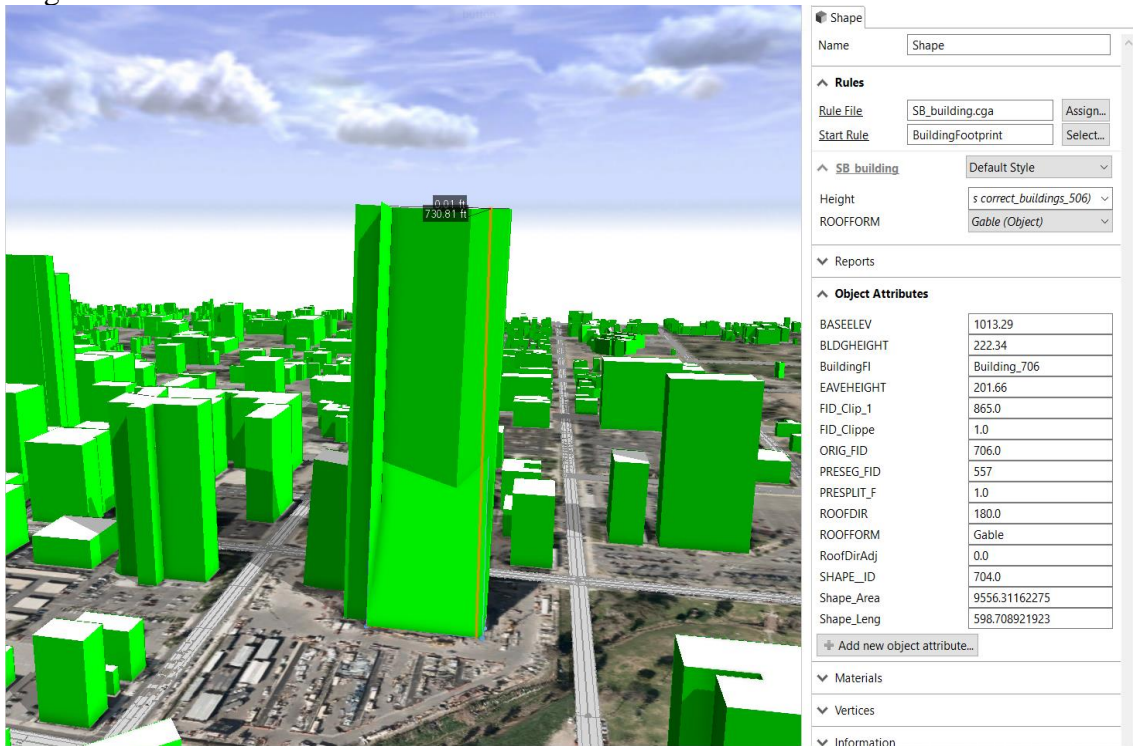


**Figure 5-15: Colorized Parcels**

## **5.4 Building Layer**

To add the building layer, the building shapefile was dragged and dropped into the scene. Next, the CGA code was written to extrude the building heights to their correct height. During extrusion the building's height was in meters and the building's original measurement was in feet. To fix this written CGA code converted meters to feet so buildings would be the correct height. See Figure 5-16 below, also see Appendix B for

height code.



**Figure 5-16: Height Difference Real Height and Height Given by The Attribute**

The remainder of the CGA code for the buildings layer was written for walls, roofs types, and facades; this helped bring to life the buildings. The completed the CGA code was applied to the Building\_From\_Footprint.cga code in the Esri.lib folder in Figure 5-17. See Appendix D for code reference.





**Figure 5-17: Facades Applied and Correct Building Height**

Upon completing the building layer, it was vital to manually go through the city and check that all buildings were level and in the correct spot. If a building was not level (Figure 5-18), the correct way to change this was through fixing the vertices by finding one vertex that was on the ground and applying that elevation to all the other corners (Figure 5-19). The vertices tab can be found in the inspector window for the selected object.



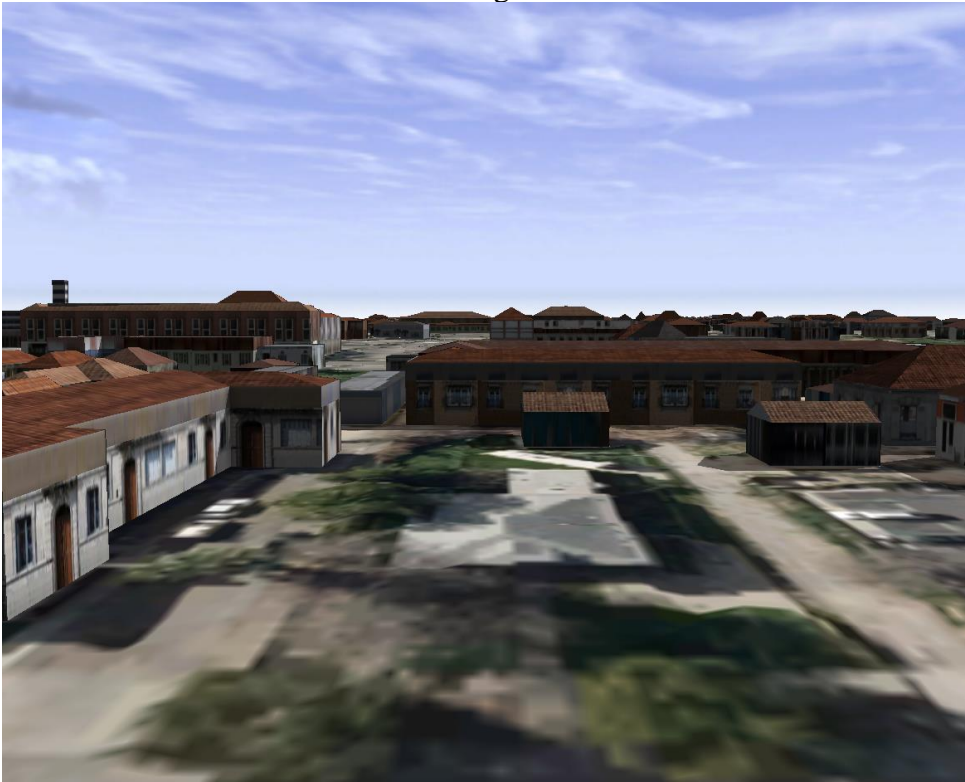
**Figure 5-18: Tilted Building**



**Figure 5-19: Corrected Tilt**

While going through the buildings layer manually, there were occasionally missing buildings this is do not having the building in the database. To remedy this, the Polygonal shape creation tool was used and the footprint of the building was outlined. Then, the

CGA code that was previously made was applied to this new structure. Figures 5-20 and 5-21 illustrate the creation of a building for where there is none.



**Figure 5-20: Missing Building**





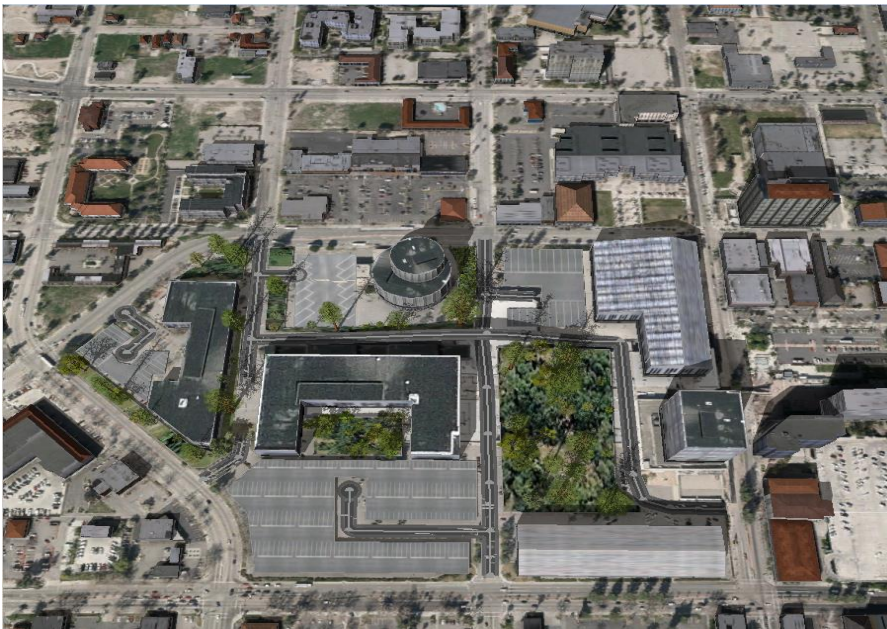
**Figure 5-21: Created Building**

## 5.5 New Development

Once the scenes layers were edited, the program created a new scenario that could house a new development, a new computer aided design (CAD) file, rivet file, SketchUp file, or a CityEngine creation. To do this, the program created a new scenario and deleted the buildings desired, as seen in this case of the Carousel Mall in San Bernardino's downtown area



**Figure 5-22: Carousel Mall Before**



**Figure 5-23: Carousel Mall After**

To create this scene, the program first created roads by using the polygonal street creation tool. Upon finishing the streets, dragged and dropped, previously used for the other street layer was applied. To create a bridge, the road going over the bridge was highlighted hovered over the gray dot and simply dragged on the Z-axis and creates a bridge-like setup. In creating vegetation, click a vegetation parcel, go to the Esri library folder, go-to rules, go to plants and dragged and dropped plant distributor rule file onto the selected parcel. This will then generate plants on to the parcel which had the ability to change over time. Select the other plant rule above the plant distributor, which could also edit and change the variation of plants in that parcel as an option. Finally, for creating the view analysis and shadow analysis, each button in the top right-hand corner was dragged and dropped to where the view and shadow analyses were performed. The results will show up in the inspector window about where the location analysis happened.

## **5.6 Summary**

In summary, to go from ArcGIS Pro to CityEngine involved creating the dataset and clipping it to the extent. Then the user performed editing and CGA coding onto the data to create a cityscape and performed editing and creation tools to make the scene. An analysis was performed on the various new developments to make sure that the new developments were correct and fitted into the environment correctly. Creation of this project was with little challenge and creating the various visual analysis where straightforward.



## Chapter 6 – Results and Analysis

Upon completion, the project was successful in Creating a 3D scene using CityEngine software with the data that the City of San Bernardino IT Department provided. However, there were many difficulties and challenges in going from 2D data to 3D data. For instance, creating footprint data with height attribute attached to it was a challenge to overcome. Using the data that was provided from the city, another challenge was to clean up the street layer data and have an accurate, correct, and current look and feel of the streets of downtown San Bernardino. The view analysis results, however, were a success with the new building development site that the city asked for. All view analyses were successful in running and testing the overall data and feel of the city in the CityEngine software and was a success in client satisfaction. The following results and challenges listed above are detailed in this chapter.

### 6.1 CityEngine Scene

This project had three portions of completion in the results section: creating the CityEngine scene to accurately represent the downtown area of San Bernardino, creating a new development area somewhere in the city, such as the Carousel Mall, and view analysis and shadow analysis on the new development site to make sure that new developments were in line with the stakeholders initial plan, and there were no drastic obstructions to other existing structures in the area. Although the client from the city of San Bernardino only requested to have his 2D data converted into 3D data, the project was extended into having a new development created in CityEngine software, that development being in the Carousel Mall area. The client also requested to have view analysis and shadow analysis performed in the newly developed area. See Figures 6-1 and 6-2.



**Figure 6-1: CityEngine Scene Overview**



**Figure 5-2: CityEngine Scene Detailed View**

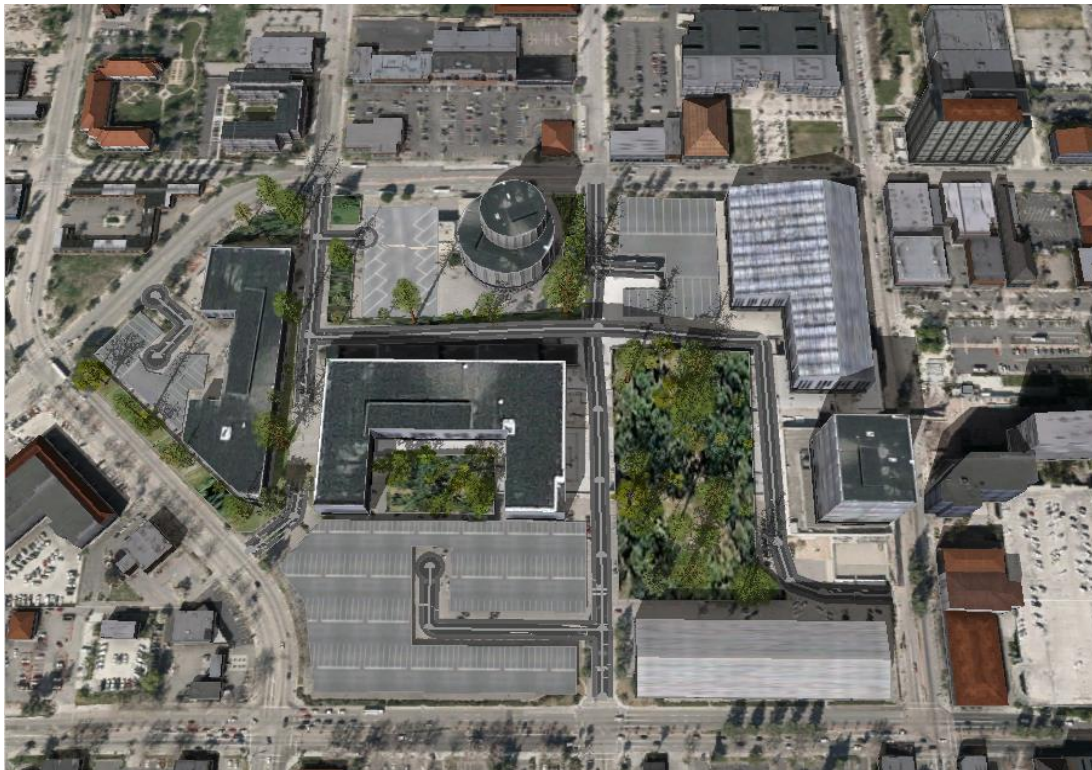
## **6.2 Development**

With completing the CityEngine scene to specifications, the client wanted time to redesign and redevelop the Carousel Mall. With this came relatively easy and straightforward workflows; the creation of the new development took approximately three hours from knowing very little on the subject matter to creating the cityscape, street networks, vegetation, and bridge. Learning how to create from nothing to then editing various parts of the newly developed scene was relatively straightforward with little to no problems. The only challenge in creating this new development area was doing design for the various potential uses of this parcel area. See Figures 6-3, 6-4, 6-5, and 6-6.





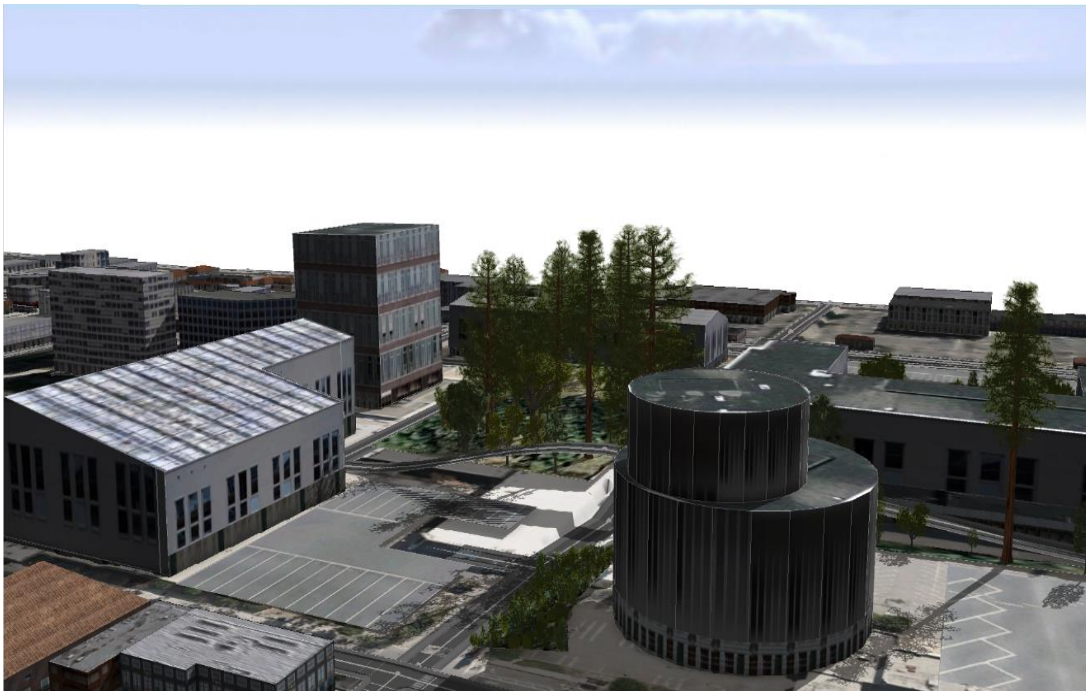
**Figure 6-3: Carousel Mall - Before**



**Figure 6-4: Carousel Mall - After**



**Figure 6-5: Carousel Mall Before Detailed**



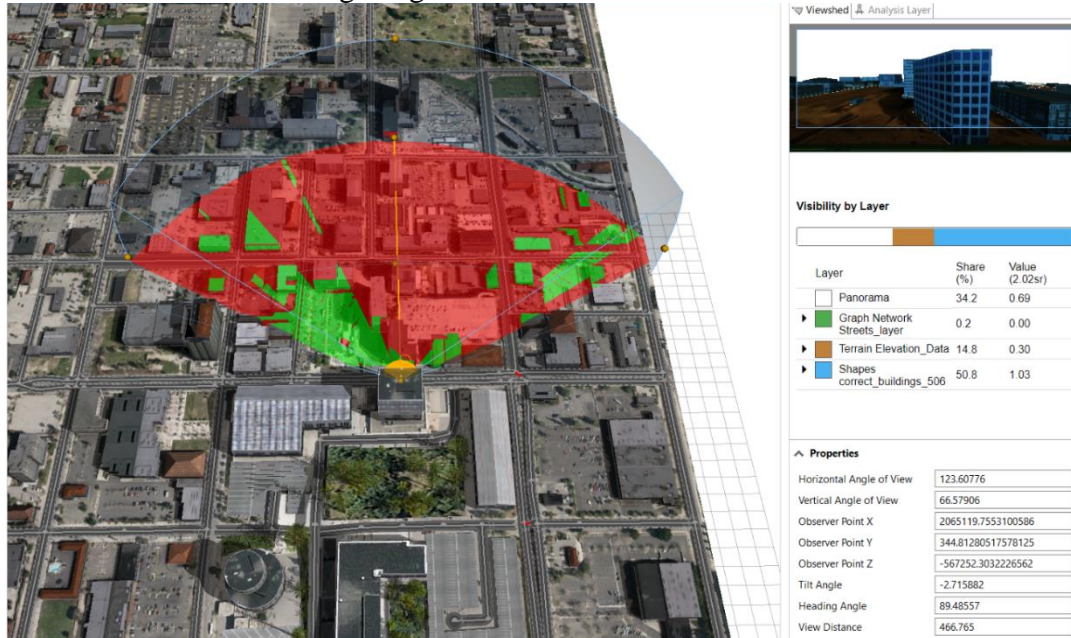
**Figure 6-6: Carousel Mall After Detailed**

### **6.3 View Analyses**

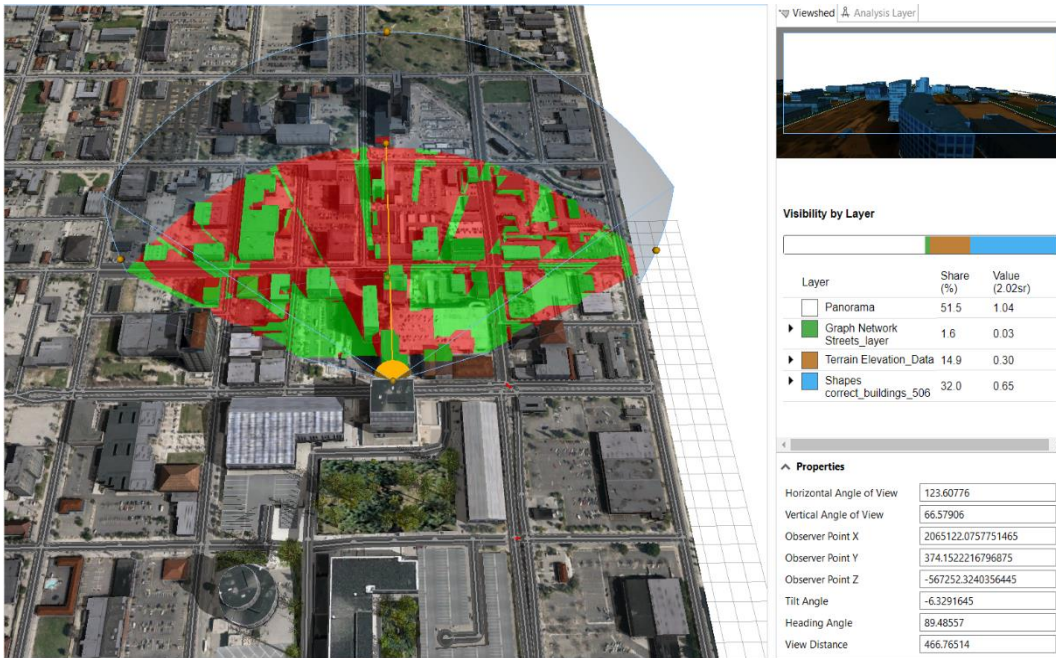
View analyses and the shadow analyses were successful, and they can demonstrate the changes visually and effectively. The inspector view depicts how much percentage of each type of asset one person could see from a fixed vantage point or focal point. The task was straightforward, with little to no problem to occur during the analyses. The



viewfinder shows the view range; green represents what is seen from the fixed focal point on the building, and red represents what cannot be seen from that focal point. On the right-hand side of the images, you can see at the top of the viewfinder what it looks like to be at that angle below the viewfinder, and the visibility layer depicts a breakdown of the percentage one can see of each type of asset (i.e. sky(panorama), buildings, streets, terrain), from that focal point. Lastly, in the properties section on the right-hand side, you can physically adjust the observation point, vertical angle view, horizontal angle view, tilt angle, heading angle, and view distance manually, or go into the main view finder and move the current extent of the viewshed tool. Figure 6-7 and 6-8 illustrate viewshed creation with a low and high angle to show the view difference on the same building.



**Figure 6-7: Viewshed Creation Low Angle**



**Figure 6-8: Viewshed Creation High Angle**

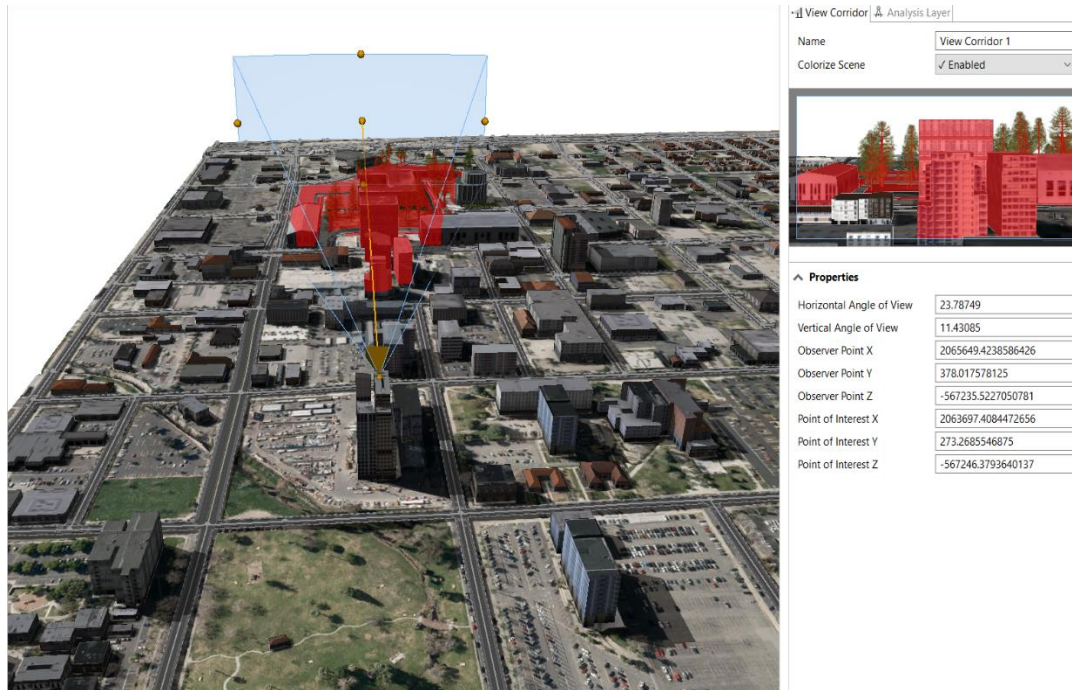
The Dome analysis is very similar to the viewshed analysis in creation and results. On the right-hand side in the inspector viewed window similar to viewshed there is an image of the 360 degree's scenes in the top right. The visibility by layer is similar to the viewshed analysis, and the properties tab on the bottom right is similar to the viewshed analysis in functionality. In Figure 6-9 the image illustrates the View Dome Creation tool that shows the viewer a 360° view from that vantage point.



**Figure 6-9: View Dome Creation**



Next is the view corridor analysis tool, and this tool establishes views from surrounding buildings near and far to the new development. It is best to run this analysis after you have performed the creation of your development or developments. Figure 6-10 and 6-11 shows a building with shadow alteration preformed on it.



**Figure 6-10: View Corridor**

Similar to dome and viewshed analyses, the view corridor analysis tool is only exception is that it does not have a visibility bilayer breakout. The view corridor tool has just a view finder window from the position chosen and a properties layer where you can critique and adjust the image accordingly.

Lastly there is the shadow analysis and to perform this go to the scenarios icon and then adjust the settings to your desired preference for a building to see how the building changes from morning to dusk. There are also ways you can change the solar intensity and the azimuth, if needed and desired for the creation of your new development. Figures 6-11 and 6-12 are illustrating the effects of shadows on a building.



**Figure 6-11: Morning Shadow**



**Figure 6-12: Afternoon Shadow**

## **6.4 Summary**

The results were successful in creating a 3D cityscape of downtown San Bernardino with a new development and applying various visual analyses. City of San Bernardino needs to have their own building with height data to accurately portray their city and include vegetation data to give their city more detail and realistic accuracy to reality. This project is going to be best utilized when more developments and city projects come to having a the

## **Chapter 7 – Conclusions and Future Work**

### **7.1 Conclusion**

The City of San Bernardino initially sought to create a CityEngine scene for its downtown area. This, in turn, would help its urban planners and all stakeholders have a better, more accurate, and clearer understanding of what will happen to the downtown and surrounding area. With this, they were asking to have a 3D model created from their 2D buildings, streets, parcels, and a terrain layer of the downtown area. The process that was followed for this project was to compile the datasets in ArcGIS Pro, attach height attributes to the building footprints (found online on [arcgisonline.com](http://arcgisonline.com)), and clip the data to the extent of the downtown area. Once clipped, the datasets were converted into shapefiles using the Feature Class To Shapefile tool and imported into a CityEngine scene. CGA code and editing tools and techniques were then used to further clean and improve the data. CityEngine tools were used to create a potential development of several new buildings and streets. View and shadow analyses were then conducted on the new buildings to determine their impact on adjacent buildings and surrounding blocks.

### **7.2 Future Work**

Future work could include adding vegetation to the CityEngine scenes. Vegetation would provide a more natural feeling, greater detail, and more realistic accuracy when performing view and shadow analyses.

Another area that could be pursued is to import the CityEngine file into the Unity or Unreal game engines in order to simulate the scene in a virtual reality (VR) environment. Stakeholders would be able to visualize the proposed developments by walking down the streets and around the buildings, and even positioning themselves at different vantage points within the buildings. Also, within the VR environment, it is possible to visualize changes in the weather and lighting conditions to simulate views at different times of day and different seasons.

Another would be digitizing and creating the rest of the San Bernardino city scape and then ultimately creating the rest of the County.

The use of CityEngine could benefit the stakeholders with a clearer and more effective visualization of the future of their city.





## Works Cited

- Biljecki, F., Stoter, J., Ledoux, H., Zlatanoza, S., & Coltekin, A. (2015). Applications of 3D City Models: State of the Art Review. (W. Kainz, Ed.) *ISPRS International Journal of Geo-Information*, 2843-2889.
- City of San Bernardino - About. (n.d.). Retrieved June 26, 2019, from <http://www.sbcity.org/about/default.asp>
- Esri.com. (2018). *Esri CityEngine | Advanced 3D City Design Software*. [online] Available at: <https://www.esri.com/en-us/arcgis/products/esri-cityengine/overview> [Accessed 18 Oct. 2018].
- Kunze, A., Burkhard, R., Gebhardt, S., & Tuncer, B. (2012). *Visualization and Decision Support Tools in Urban Planning*. Berlin Heidelberg: Springer-Verlag.
- Ranzinger, M., & Gleixner, G. (1997). GIS Datasets for 3D Urban Planning. *Computers, Environment, and Urban Systems*, 21(2), 159-173. Retrieved from <https://redlands.illiad.oclc.org/illiad/illiad.dll?Action=10&Form=75&Value=37129>
- Rapid Prototyping | definition in the Cambridge English Dictionary. (2019). Retrieved 9 August 2019, Retrieved from <https://dictionary.cambridge.org/us/dictionary/english/rapid-prototyping>
- Schobesberger, D., & Patterson, T. (2006). Evaluating the Effectiveness of 2D vs. 3D Trailhead Maps. 6<sup>th</sup> ICA Mountain Cartography Workshop 201-205.
- Shiode, N. (2000). 3D urban models: Recent developments in the digital modelling of urban environments in three-dimensions. *GeoJournal*, 52(3), 263-269.
- Singh, S P, Jain, K., & Mandla, V. R. (2014). Image-based Virtual 3D Campus modeling by using CityEngine. *American Journal of Engineering Science and Technology Research* 2(1), 1-10.
- Stoter, J., & Zlatanova, S. (2009). *3D GIS, where are we standing?* Unpublished manuscript, Section GIS Technology, Delft University of Technology, The Netherlands.
- U.S. Census Bureau (2019) QuickFacts: San Bernardino City, California. Retrieved June 26, 2019, from <https://www.census.gov/quickfacts/sanbernardinocitycalifornia>
- Wassim, S., Thierry, J., & Eric, F. (2011, January). *3D Urban Visibility Analysis with Vector GIS Data*. Unpublished manuscript, Université Jean Monnet - Saint-Etienne, 1-13.
- Zlatanova, S., Rahman, A. A., & Pilouk, M. (2002). 3D GIS: Current Status and Perspectives. Symposium on Geospatial Theory, Processing and Applications, Ottawa.

## **Appendix A. Footprint with Height Workflow**

<https://solutions.arcgis.com/local-government/help/local-government-scenes/get-started/create-schematic-scene/>

## Appendix B. Coloring Parcels using CGA code

```
/**
 * File:      Parcel_Color.cga
 * Created: 11 Jun 2019 01:00:27 GMT
 * Author:   Max Babcock
 */
version "2018.1"
@Range (0,999)
attr TYPEUSER = " "

attr TYPEUSE = 0

@StartRule
getColorPerCatorgyTYPEUSE() -->
    print(TYPEUSE / 100 % 1)
    #by dividing TYPEUSE by this creates a hue for each number range
the % is modulus -
    #operator which a gives you the residual of the division.
    color(TYPEUSE / 1000, TYPEUSE / 100 % 1, TYPEUSE / 10 % 1)
```

## Appendix C. Building footprint code

```
/**
 * File:      SB_building.cga
 * Created:   7 May 2019 17:00:17 GMT
 * Author:    Max
 * Modified:  Chris Wilkins
 */

version "2018.1"
*/@Range (0,200)
attr Height = 0
const _Height = meters(Height)
@Range("Feet", "Meters")
attr Units = "Feet"
meters(distance) = case Units == "Feet": distance * 0.328084 else:
distance
@Range ("Gable","Hip","Flat")
attr ROOFFORM = " "
@Range("Texture")
attr DisplayType = "Texture"
attr WallTexture = fileRandom("/my_city/assets/walltextures/*")
attr SlopedRoofTexture = fileRandom("/my_city/assets/slopedroof/*")
attr FlatRoofTexture = fileRandom("/my_city/assets/flatroof/*")
@StartRule
BuildingFootprint -->
    ExtrudeMass

ExtrudeMass -->
    extrude(_Height)
    comp (f) {top:Roof|side:Wall}

Wall -->
    Texture("Wall")

Texture (whichFace) -->
    case DisplayType == "Texture":
        case whichFace == "Wall":
            setupProjection(0, scope.xy,4,5)
            projectUV(0)
            texture(WallTexture)
        case whichFace == "SlopedRoof":
            setupProjection(0, scope.xy,scope.sx,scope.sy)
            projectUV(0)
            texture(SlopedRoofTexture)
        case whichFace == "FlatRoof":
            setupProjection(0, scope.xy,scope.sx,scope.sy)
            projectUV(0)
            texture(FlatRoofTexture)
        else: FinalShape.
    else: FinalShape.

Roof -->
    case ROOFFORM == "Gable" : GableRoof
    case ROOFFORM == "Hip" : HipRoof
```

```

    case ROOFFORM == "Flat" : FlatRoof
    else:PrintRoof

GableRoof -->
    roofGable(30,.03,.03)
    comp (f) {aslant:Texture("SlopedRoof") | side: Texture ("Wall")}
FlatRoof -->
    extrude(.05)
    comp (f) {top:Texture("FlatRoof") | side: Texture ("Wall")}
HipRoof -->
    roofHip(45,.05)
    comp (f) {aslant:Texture("SlopedRoof") | side: Texture ("Wall")}
PrintRoof -->
    print("My roof is " + ROOFFORM)

```

## Appendix D. Building footprint code from Esri.lib

```
* File:      Building_From_Footprint.cga
* Created: 20 Mar 2011 16:42:13 GMT
* Author:   Esri R&D Center Zurich
* Modified: Max Babcock
*/

version "2018.0"

@Hidden(Usage, BuildingHeight, UpperfloorHeight)
import Facade_Textures:"/ESRI.lib/rules/Facades/Facade_Textures.cga"
(BuildingHeight=Eave_Ht, UpperfloorHeight=Floor_Ht*unitScale, Usage=Usage)
@Hidden(Usage, UpperfloorHeight)
import Facade_Schematic:"/ESRI.lib/rules/Facades/Facade_Schematic.cga"
(UpperfloorHeight=Floor_Ht*unitScale, Usage=Usage)
import Roof_Textures:"/ESRI.lib/rules/Roofs/Roof_Textures.cga"

#####
# Attributes
#
#this attr connects BLDGHeight to this rules attributes. The number 20
is a place holder you can use any number you like.
attr BLDGHEIGHT = 20
attr EAVEHEIGHT = 20

attr ROOFORM = ""

roofForm(roof) =
    case roof == "Gable" : "gable"
    case roof == "Hip"   : "hip"
    case roof == "Flat"  : "flat"
    else:"flat"

@Group("Building Settings",1)
#add BLGHeight to the default value
@Order(1) @Range(1,400) @Description("Distance from ground to bottom of
roof")
attr Eave_Ht = case ROOFORM == "Flat": BLDGHEIGHT * .328084
              else: EAVEHEIGHT *
.328084#_getInitialEaveHeight
@Order(2) @Range(1,400) @Description("Distance from ground to top of
roof")
attr Ridge_Ht = BLDGHEIGHT * .328084#_getInitialRidgeHeight
@Order(3)
@Range("Random","Agricultural","Assembly","Educational","Industry","Mer
cantile","Office","Other","Public","Residential","Service","Transport",
"Unknown","Utility")
attr Usage = _getInitialUsage
@Order(4) @Range("extrusion","setback top","setback facade","setback
base","setback everywhere")
attr Building_Form = _getInitialBuildingForm
```

```

@Order(5) @Range("flat","shed","pyramid","gable","hip","half-
hip","gablet","gambrel","mansard","gambrel-flat","mansard-
flat","vault","dome","saltbox","butterfly")
    # gable & shed combinations
attr Roof_Form = roofForm(ROOFFORM)#_getInitialRoofForm
@Order(6) @Range(2.9,5.2) @Description("in Meters")
attr Floor_Ht = 3.7
@Hidden
attr Roof_Ht = (Ridge_Ht - Eave_Ht) * unitScale

@Group("Visualization Options",2)

@Order(1) @Range("realistic with facade textures","schematic
facades","solid color")
attr Representation = "realistic with facade textures"
@Order(2) @Range(0,1)
attr Transparency = 0
@Order(3) @Color
attr OverwriteColor = "#ffffff"
@Order(4) @Color
attr RoofColor = OverwriteColor

@Group("Rule Options")

@Order(2) @Range("Meters","Feet") @Description("Units of height
attributes")
attr Unit = "Meters"

@Order(3) @Range("None","All") @Description("Report information")
attr Reporting = "None"

#####
# Consts
#

# user-driven constants
const unitScale = case Unit=="Feet": 1/0.3048006096012192 else: 1

# for curved roofs such as dome or vault
const curvedAngleResolution = 10

#####
# Functions
#

# for curved roofs such as dome or vault
calcSegmentHt(n) = Roof_Ht * (cos(n*curvedAngleResolution) -
cos((n+1)*curvedAngleResolution))

_getInitialBuildingForm =
    case Eave_Ht*unitScale < 50 : "extrusion"
    case Eave_Ht*unitScale > 100: "setback everywhere"

```



```

        else                                : 5%:"extrusion" 15%:"setback top"
15%:"setback facade" 15%:"setback base" else:"setback everywhere"

_getInitialUsage =
    case Eave_Ht>30: "Random" else: 80%:"Residential" else:"Random"

_getInitialEaveHeight =
    case geometry.area < 100 : geometry.area/rand(5,10)
    case geometry.area < 1000: geometry.area/rand(15,25)
    case geometry.area < 7000: geometry.area/rand(10,25)
    else                        : geometry.area/rand(70,200)

_getInitialRidgeHeight =
    case Eave_Ht<30: Eave_Ht+rand(3,6) else: Eave_Ht

_getInitialRoofForm =
    case Ridge_Ht < Eave_Ht+1: "flat"
    else: 40%: "hip" 50%: "gable" else: "gambrel"

#####
#####
#
# RULES
#
#####
#####

@StartRule
Generate -->
    cleanupGeometry(all,1)
    alignScopeToAxes(y) s('1,0,'1)          # make it horizontal
i.e. scale it flat
    alignScopeToGeometry(yUp, 0, longest)
    set(Eave_Ht,Eave_Ht*unitScale)
    set(Floor_Ht,Floor_Ht*unitScale)
    Reports_Lot
    set(material.opacity,1-Transparency)
    color(OverwriteColor)
    Footprint

#####
# Building Mass
#

Footprint -->
    case scope.sz < 10 || scope.sx < 10:
        Extrusion(Eave_Ht,true,1)
    case Building_Form == "setback top":
        SetbackTop
    case Building_Form == "setback facade":
        SetbackFacade
    case Building_Form == "setback base":
        SetbackBase

```

```

    case Building_Form == "setback everywhere":
        SetbackAll
    else:
        Extrusion(Eave_Ht, true, 1)

SetbackTop -->
    split(x) { 'rand(0.1, 0.3): Extrusion(Eave_Ht-
rint(rand(3)) * Floor_Ht, false, 4)
            | ~1 : Extrusion(Eave_Ht, true, 6)
            | 'rand(0.1, 0.3): Extrusion(Eave_Ht-
rint(rand(3)) * Floor_Ht, false, 4) }

SetbackFacade -->
    split(z) { 'rand(0.03, 0.2):
Extrusion(Eave_Ht * rand(0.2, 0.8), false, 2)
            | ~1 : Extrusion(Eave_Ht, true, 6)
            | 'rand(0.03, 0.2):
Extrusion(Eave_Ht * rand(0.2, 0.8), false, 2) }

SetbackBase -->
    [ extrude(3 * Floor_Ht) Mass(false) ]
    t(0, 3 * Floor_Ht, 0)
    split(x) { 'rand(0.6, 0.8): Extrusion(Eave_Ht - 3 * Floor_Ht, true, 6) }

SetbackAll -->
    [ extrude(3 * Floor_Ht) Mass(false) ]
    t(0, 3 * Floor_Ht, 0)
    set(Eave_Ht, Eave_Ht - 3 * Floor_Ht)
    split(x) { 'rand(0.6, 0.8):
        split(z) { '0.2: Extrusion(Eave_Ht * rand(0.2, 0.8), false, 2)
            | ~1 : SetbackTop
            | '0.2:
Extrusion(Eave_Ht * rand(0.2, 0.8), false, 2) }
    }

Extrusion(height, constructRoof, maxLength) -->
    convexify(maxLength)
    comp(f) { all: alignScopeToGeometry(yUp, 0, longest)
ExtrusionConvexified(height, constructRoof, maxLength) }

ExtrusionConvexified(height, constructRoof, maxLength) -->
    case scope.sx < maxLength + 1 || scope.sz < maxLength + 1: NIL
    else:
        Reports_GFA(height)
        extrude(height) Mass(constructRoof)

Mass(constructRoof) -->
    case constructRoof:
        comp(f) { side : Facade | top : Roof }
    else:
        comp(f) { side : Facade | top : RoofPlane }

#####
# Roof Generation
#

```

```

Roof -->
  case Roof_Ht == 0 : RoofPlane
  case Roof_Form == "shed" : ShedRoof
  case Roof_Form == "pyramid" : PyramidRoof
  case Roof_Form == "gable" : GableRoof
  case Roof_Form == "hip" : HipRoof
  case Roof_Form == "half-hip" : HalfHipRoof
  case Roof_Form == "gabled" : GabledRoof
  case Roof_Form == "gambrel" : GambrelRoof
  case Roof_Form == "mansard" : MansardRoof
  case Roof_Form == "gambrel-flat" : GambrelFlatRoof
  case Roof_Form == "mansard-flat" : MansardFlatRoof
  case Roof_Form == "vault" : VaultRoof
  case Roof_Form == "dome" : DomeRoof
  case Roof_Form == "saltbox" : SaltboxRoof
  case Roof_Form == "butterfly" : ButterflyRoof
  else : FlatRoof

# basic roof types

ShedRoof -->
  roofShed(15) RoofMassScale

GableRoof -->
  roofGable(45,0,0,false,0) RoofMassScale

HipRoof -->
  roofHip(45) RoofMassScale

PyramidRoof -->
  roofPyramid(45) RoofMassScale

# gable & hip combinations

HalfHipRoof -->
  roofGable(45,0,0,false,0) s('1, Roof_Ht, '1) # creates a gable
  roof and sets its height to the given roof height
  split(y) { '0.5: RoofMass(true) # ...
  comp(f) { bottom: NIL | horizontal:
set(Roof_Ht, Roof_Ht*0.5) HipRoof } } # ... and invokes a hip roof on
the top

GabledRoof -->
  roofHip(45) s('1, Roof_Ht, '1)
  split(y) { '0.5: RoofMass(true)
  comp(f) { bottom: NIL | horizontal:
set(Roof_Ht, Roof_Ht*0.5) GabledRoof } }

# gable/hip double-pitched

GambrelRoof -->
  roofGable(70,0,0,false,0)
  split(y) { Roof_Ht*0.7: RoofMass(true)
  comp(f) { bottom: NIL |
horizontal: set(Roof_Ht, Roof_Ht*0.3) GableRoof } }

```

```

MansardRoof -->
    roofHip(70)
    split(y){ Roof_Ht*0.7: RoofMass(true)
                                     comp(f){ bottom: NIL |
horizontal: set(Roof_Ht,Roof_Ht*0.3) HipRoof } }

# gable/hip with flat top

GambrelFlatRoof -->
    roofGable(45,0,0,false,0)
    split(y){ Roof_Ht: RoofMass(false) }

MansardFlatRoof -->
    roofHip(45)
    split(y){ Roof_Ht: RoofMass(false) }

# round roofs

VaultRoof -->
    VaultRoof(90/curvedAngleResolution-1)

VaultRoof(n) -->
    case n > 0: roofGable(n*curvedAngleResolution,0,0,false,0)
               split(y){ (calcSegmentHt(n)): RoofMass(n!=1)

comp(f){ bottom: NIL | horizontal: VaultRoof(n-1) } }
    else: NIL

DomeRoof -->
    DomeRoof(90/curvedAngleResolution-1)

DomeRoof(n) -->
    case n > 0: roofHip(n*curvedAngleResolution)
               split(y){ (calcSegmentHt(n)): RoofMass(n!=1)

comp(f){ bottom: NIL | horizontal: DomeRoof(n-1) } }
    else: NIL

# gable & shed combinations

SaltboxRoof -->
    roofShed(45) s('1,1.5*Roof_Ht,'1)
    split(y){ '0.333: RoofMass(true)
                                     comp(f){ bottom: NIL | horizontal:
set(Roof_Ht,Roof_Ht*0.5) roofGable(45,0,0,false,geometry.nVertices-1)
RoofMassScale } }

ButterflyRoof -->
    split(y){ '0.5: roofShed(45,geometry.nVertices/2) RoofMassScale |
'0.5: ShedRoof }

# flat roof

FlatRoof -->
    case Roof_Ht > 0.1:
        RoofPlane offset(-0.4,border) extrude(Roof_Ht)
RoofMass(false)

```

```

        else:
            RoofPlane

# roof volume

RoofMassScale -->
    s('1,Roof_Ht,'1)
    RoofMass(false)

RoofMass(removeBottomAndTop) -->
    case removeBottomAndTop:
        comp(f){ horizontal: NIL | vertical: Facade | all:
RoofPlane }
    else: # remove only the bottom face
        comp(f){ bottom: NIL | vertical: Facade | all: RoofPlane }

#####
# Surface Texturing & Coloring
#

RoofPlane -->
    case Representation == "realistic with facade textures":
        Roof_Textures.Generate
    else:
        color(RoofColor)

Facade -->
    case Representation == "realistic with facade textures":
        Facade_Textures.Generate
    case Representation == "schematic facades":
        case OverwriteColor == "#ffffff":
            Facade_Schematic.Generate
        else:
            set(Facade_Schematic.SecondaryColor,OverwriteColor)
            Facade_Schematic.Generate
    else:
        color(OverwriteColor)

#####
# Reports
#

Reports_Lot -->
    case Reporting=="All":
        report("Footprint Area (m2)",geometry.area)
        report("Nbr of Floors",rint(Eave_Ht/Floor_Ht))
        NIL
    else:
        NIL

# height      height of extruded part
Reports_GFA(height) -->
    case Reporting=="All":

```

```
        report("Gross Floor Area  
(m2)", geometry.area*rint(height/Floor_Ht))  
        NIL  
    else:  
        NIL
```