Dataset Description for:

Aerial imagery object identification dataset for building and road detection, and building height estimation

Dataset Authors (listed alphabetically): Kyle Bradbury, Benjamin Brigman, Leslie Collins, Timothy Johnson, Richard Newell, Sebastian Lin, Sophia Park, Sunith Suresh, Hoël Wiesner, Yue Xi

The 2016 Data Plus Energy Analytics Group assembled a dataset that can be readily used for object recognition techniques. This dataset can specifically be used as a ground-truth for finding the location of buildings and roads, and the estimation of building height. This documentation includes (1) a detailed description of the dataset and suggestions on how it may be used, (2) the process by which the data were processed and compiled into their current form, and (3) summary statistics of the data.

Energy researchers and resource planners can gain valuable information about geospatial distribution of energy needs by using building volume as a rough proxy for demand. To determine the volume of a building from a high-resolution aerial image using computer vision algorithms requires first detecting a building in an image, then estimating the perimeter of its base and finally its height (potentially using information from the building's shadow). This dataset contains building footprints and height information for over 44,000+ buildings in varied environments which will serve as ground truth to test and train computer vision algorithms. The features contained within this data set (building & road footprints, landscape height from LIDAR) can be used to aid a variety of research efforts from object detection in remote sensed images to 3D modeling of cities.

Contents

Data Description

Overview

Object detection in images utilizes computer vision algorithms which require large datasets to train effectively. This project compiled a dataset comprised of 25 instances of high-resolution aerial orthoimagery, object footprints (buildings & roads) and landscape topology data for nine urban and suburban locations across the United States. The dataset was created by collating information from three separate sources and processing the data into a format readily readable by MATLAB. Aerial orthoimagery (0.15m to 0.3m resolution) was obtained from the US Geological Survey (USGS). Building footprint and road annotations corresponding to the image was obtained from the Open Street Maps (OSM). Point clouds of height values for the landscape corresponding to the image were obtained from LIDAR surveys collected by various state and federal agencies. The dataset was cleaned, processed, and uploaded to online digital data repository Figshare. Each image subset contains the files outlined in Figure 1.

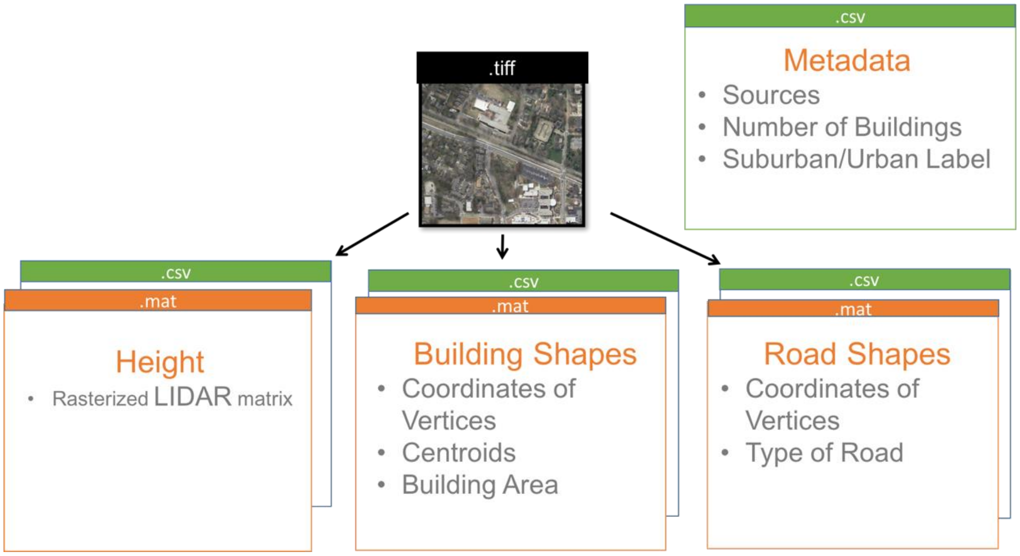


Figure 1 - Overview of information contained in the final dataset

Data Sources

(A) USGS High Resolution Orthoimagery: aerial imagery provided freely by the government. Resolution ranges from 0.15m to 0.3m. Images are collected by individual states and subcontractors, who have to meet a certain set of USGS format requirements. Orthoimagery undergoes a number of processing steps to correct for lens distortion, remove clouds, and make image color uniform. All high-resolution orthoimages include 3 visible light bands (Red, Green, Blue), with many also having a 4th near infrared color layer. This is the .tiff file.

U.S. Geological Survey (2012-2015), *USGS Products* at http://earthexplorer.usgs.gov

(B) LIDAR height: To find the height of objects in the image, we use LIDAR data collected by state and local agencies and provided by USGS, NOAA, or state GIS agencies. LIDAR is topographic height information collected by planes shooting lasers at the ground. It comes in a binary point cloud file containing x, y, and z values. Average point spacing is 0.5m, though it is not uniform. Related files contain “height” in the name.

NOAA (2010-2016), *Elevation Data* at https://coast.noaa.gov
Texas Natural Resources Information System (2012), *CAPCOG* at https://tnris.org/data-catalog/entry/capcog-2012-140cm

(C) Open Street Map (OSM) shapefiles: OSM is an online collaborative mapping project that lists millions of roads, buildings, parks, and waterways around the world. Their building footprint outlines are in many cases hand drawn by users, or created from government agency building footprint maps. Roads are created in a similar fashion. We download this building footprint and road information for selected areas as shapefiles to make processing in ArcGIS and MATLAB easier. Related files contain “building” or “road” in the name.

Open Street Map (2016), *Map Data* at http://www.openstreetmap.org

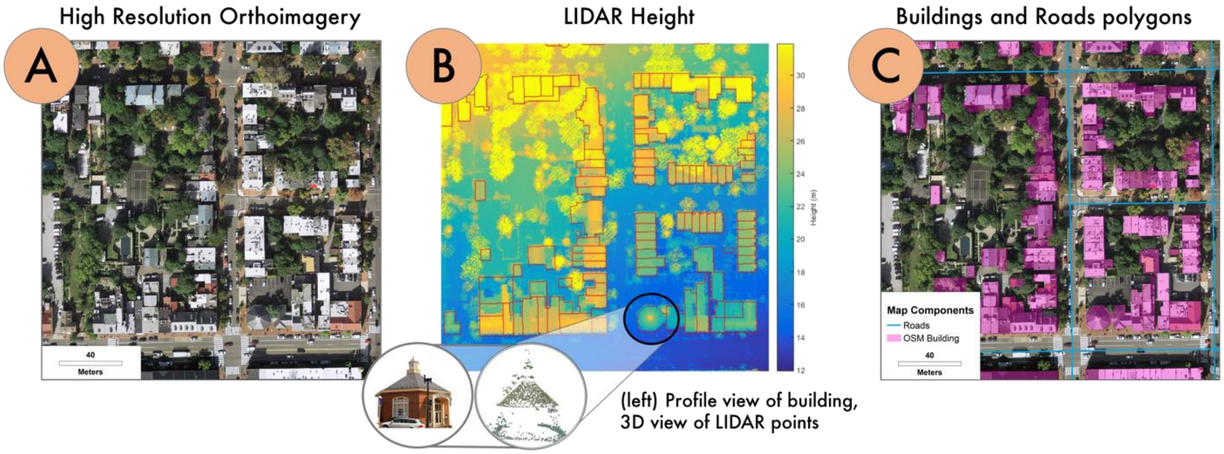


Figure 2 - Visualization of final output data

Description of Included Files

The provided dataset includes 25 unique images from 9 different cities, each with its own set of building, road, and height information. Images and their associated files are identified consistently throughout with each file listing the city name and image number: “city_#.” For example, the MATLAB readable height data for the first image in Arlington, MA is called “Arlington_01_height.mat.” Map of image locations is shown in Figure 3.

Image Name	Has IR?	Resolution	Density	N. of Buildings	City	State	Year Taken
Arlington_01	Yes	0.3m	Suburban	2232	Arlington	MA	2013
Arlington_02	Yes	0.3m	Suburban	2139	Arlington	MA	2013
Arlington_03	Yes	0.3m	Suburban	1570	Arlington	MA	2013
Atlanta_01	No	0.5ft	Suburban	302	Atlanta	GA	2013
Atlanta_02	No	0.5ft	Suburban	395	Atlanta	GA	2013
Atlanta_03	No	0.5ft	Suburban	361	Atlanta	GA	2013
Austin_01	Yes	0.5ft	Suburban	2387	Austin	TX	2012
Austin_02	Yes	0.5ft	Suburban	3412	Austin	TX	2012
Austin_03	Yes	0.5ft	Suburban	2005	Austin	TX	2012
DC_01	No	0.16m	Urban	1296	Washington	DC	2013
DC_02	No	0.16m	Urban	1006	Washington	DC	2013
NewHaven_01	Yes	0.3m	Suburban	1174	New Haven	CT	2012
NewHaven_02	Yes	0.3m	Suburban	1640	New Haven	CT	2012
NewYork_01	Yes	0.5ft	Urban	871	New York	NY	2014
NewYork_02	Yes	0.5ft	Urban	1253	New York	NY	2014
NewYork_03	Yes	0.5ft	Urban	1287	New York	NY	2014
Norfolk_01	Yes	1ft	Suburban	2053	Norfolk	VA	2013
Norfolk_02	Yes	1ft	Suburban	2079	Norfolk	VA	2013
Norfolk_03	Yes	1ft	Suburban	2158	Norfolk	VA	2013
SanFrancisco_01	Yes	0.3m	Urban	4123	San Francisco	CA	2015
SanFrancisco_02	Yes	0.3m	Urban	4186	San Francisco	CA	2015
SanFrancisco_03	Yes	0.3m	Urban	5305	San Francisco	CA	2015
Seekonk_01	Yes	0.3m	Suburban	209	Seekonk	MA	2013
Seekonk_02	Yes	0.3m	Suburban	453	Seekonk	MA	2013
Seekonk_03	Yes	0.3m	Suburban	583	Seekonk	MA	2013

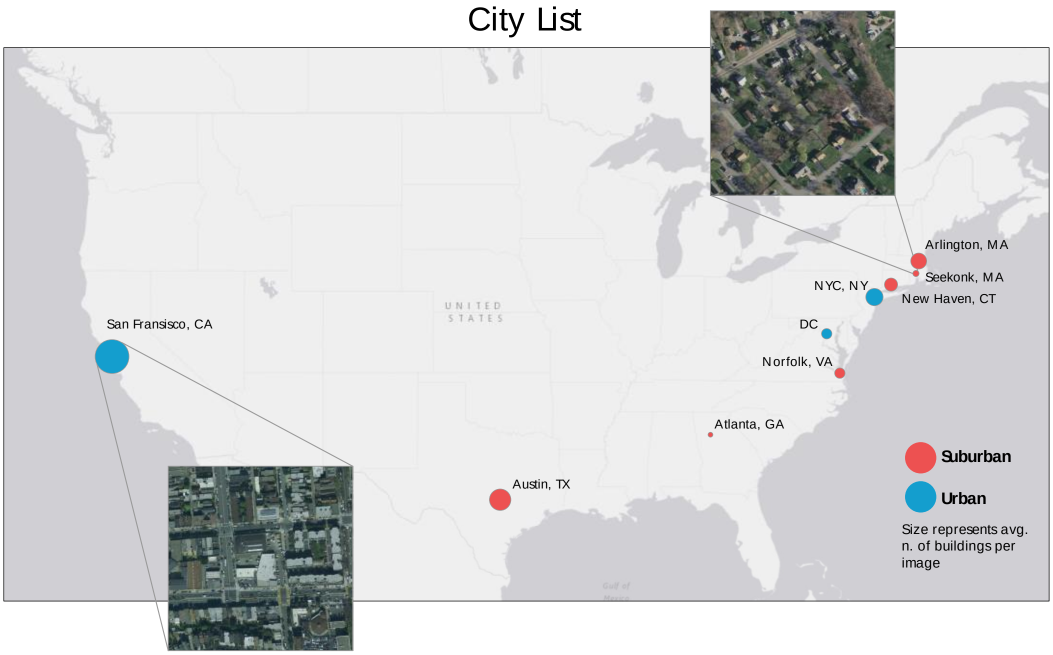


Figure 3 - Map of image locations, classification, and average number of buildings per image

Our output comprises of three components: high resolution orthoimagery, building and road shape information, and rasterized and interpolated LIDAR information. The output with all of this information is available in the files organized by city. For each image in each city, the files named and described below are available:

Files Included for Each Location

Category	Naming Convention	Description	Documentation Order
Buildings	City_#_buildingCell.mat	Contains information about building polygons in image (includes information in both City_#_buidingCoord.csv and City_#_buildings.csv). Provided as a MATLAB cell file.	1
	City_#_buildingCoord.csv	Contains location information about coordinates of vertices for each building polygon	2
	City_#_buildings.csv	Contains information about building polygons in image, without vertices coordinates	3
Heights	City_#_height.mat	Contains rasterized LIDAR matrix with height values	4
	City_#_height.csv	Contains rasterized LIDAR matrix with height values	5
Images	City_#.tif	TIFF high resolution orthoimagery	6
Roads	City_#_road_cell.mat	Contains information about road polylines in image (includes information in both City_#_roadCoord.csv and City_#_road_cell.csv	7
	City_#_roadCoord.csv	Contains location information about coordinates of vertices for each road polyline	8
	City_#_road.csv	Contains information about road polylines in image, without vertices coordinates	9

Information on the contents and parameters of each file are listed and described in the following pages in the listed documentation order in the table above.

1. City_#_buildingCell.mat

Field	Description
Image_Name	Name for corresponding high resolution orthimagery: City_#
Polygon_ID	Polygon number label for buildings in image, unique for each image
OSM_ID	Polygon ID assigned by Open Street Map
Centroid_X	X Value (pixel index) of centroid for polygon, in which centroid not necessarily inside polygon. Caculated by using linear transformation on centroid longitude
Centroid_Y	Y Value (pixel index) of centroid for polygon, in which centroid not necessarily inside polygon. Caculated by using linear transformation on centroid longitude
Centroid_Longitude	Longitude (decimal degrees NAD83) of centroid for polygon, in which centroid not necessarily inside polygon. Calculated with ARCGIS.
Centroid_Latitude	Latitude (decimal degrees NAD83) of centroid for polygon, in which centroid not necessarily inside polygon. Calculated with ARCGIS.
Area_Pixels	Number of pixels inside polygon. Calculated by transforming area in meters to number of pixels.
Area_Meters	Area in meters squared of polygon. Calculated using ARCGIS.
Polygon_X	Vector of X values (pixels) of vertices of polygon, indices correspond to Y. Calculated by using linear transformation on longitude values from shapefile. Nan values are a "pen-up" command, signifying the end of one polygon.
Polygon_Y	Vector of Y values (pixels) of vertices of polygon, indices correspond to X. Calculated by using linear transformation on longitude values from shapefile. Nan values are a "pen-up" command, signifying the end of one polygon.

2. City_#_buildingCoord.csv

Field	Description
Image_Name	Name for corresponding high resolution orthimagery: City_#
Polygon_ID	Polygon number label for buildings in image, unique for each image
Number_Vertices	Number of Vertices for polygon
[Untitled]	Rest of the row contains X,Y pairs (pixel indices) defining locations of polygon vertices. Nan values are a "pen-up" command, signifying the end of one polygon. Format is: X1 Y1 X2 Y2 .... , number of pairs is equal to Number_Vertices for polygon. Calculated by using linear transformation on longitude, latitude values from shapefile.

3. City_#_buildings.csv

Field	Description
Image_Name	Name for corresponding high resolution orthimagery: City_#
Polygon_ID	Polygon number label for buildings in image, unique for each image
OSM_ID	Polygon ID assigned by Open Street Map
Centroid_X	X Value (pixel index) of centroid for polygon, in which centroid not necessarily inside polygon. Caculated by using linear transformation on centroid longitude
Centroid_Y	Y Value (pixel index) of centroid for polygon, in which centroid not necessarily inside polygon. Caculated by using linear transformation on centroid longitude
Centroid_Longitude	Longitude (decimal degrees NAD83) of centroid for polygon, in which centroid not necessarily inside polygon. Calculated with ARCGIS.
Centroid_Latitude	Latitude (decimal degrees NAD83) of centroid for polygon, in which centroid not necessarily inside polygon. Calculated with ARCGIS.
Area_Pixels	Number of pixels inside polygon. Calculated by transforming area in meters to number of pixels.
Area_Meters	Area in meters squared of polygon. Calculated using ARCGIS.

4. City_#_height.mat

Field	Description
Zheight	Array of doubles, which is the rasterized LIDAR points. Points interpolated to fit grid using nearest neighbor inteprolation. Indices correspond to high resolution orthoimagery.

5. City_#_height.csv

Field	Description
[Untitled]	Array of doubles, which is the rasterized LIDAR points. Points interpolated to fit grid using nearest neighbor inteprolation. Indices correspond to high resolution orthoimagery.

6. City_#.tif

Field	Description
Tiff Image	Geotiff high resolution orthimagery. Contains 3-4 channels: Red, Green, Blue, Color Near-Infrared (if 4).

7. City_#_road_cell.mat

Field	Description
Image_Name	Name for corresponding high resolution orthimagery: City_#
Polyline_ID	Polyline number label for roads in image, unique for each image
Road_Name	Official name for road. ex) Campus Drive
OSM_ID	Polyline ID assigned by Open Street Map
Type	Subclassification of road. ex) Highway
Road_X	Vector of X values (pixels) of vertices of polyline, indices correspond to Y. Calculated by using linear transformation on longitude values from shapefile. Nan values are a "pen-up" command, signifying the end of one polyline.
Road_Y	Vector of Y values (pixels) of vertices of polyline, indices correspond to X. Calculated by using linear transformation on longitude values from shapefile. Nan values are a "pen-up" command, signifying the end of one polyline.

8. City_#_roadCoord.csv

Field	Description
Image_Name	Name for corresponding high resolution orthimagery: City_#
Polyline_ID	Polyline number label for roads in image, unique for each image
Number_Vertices	Number of Vertices for polyline.
[Untitled]	Rest of the row contains X,Y pairs (pixel indices) defining locations of polyline vertices. Nan values are a "pen-up" command, signifying the end of one polyline. Format is: X1 Y1 X2 Y2 .... , number of pairs is equal to Number_Vertices for polyline. Calculated by using linear transformation on longitude, latitude values from shapefile.

9. City_#_road.csv

Field	Description
Image_Name	Name for corresponding high resolution orthimagery: City_#
Polyline_ID	Polyline number label for roads in image, unique for each image
Road_Name	Official name for road. ex) Campus Drive
OSM_ID	Polyline ID assigned by Open Street Map
Type	Subclassification of road. ex) Highway

File types

There are a number of file types mentioned in this document, so here is a brief description of each:

.tiff: The Tagged Image File Format (TIFF) stores raster graphic images. In our dataset, the TIFF files are of a specific type of TIFF format called GeoTIFF, which allows for incorporation of georeferencing information such as map projections and coordinate systems.
.shp: The shape format stores information describing vector features such as points, lines and polygons. These features have attributes such as centroids, area, name etc attached to them via an accompanying file with an attribute format with a .dbf extension. These features are also indexed via another accompanying file with a shape index format with a .shx extension.
.las: The .las file format stores LIDAR point cloud information
.txt: The .txt file stores text information
.csv: A .csv file contains data separated by commas.
.mat: The .mat file holds Matlab variables and their values

Datum and Projection

Dealing with geospatial data from different sources can often lead to increased error and frustration because of inconsistent datum and projection. When working with images from USGS, with shapefiles from OSM, and LIDAR from NOAA, the data need to be re-projected to match one another. All data that was not already in NAD83 was converted into this datum. OSM shapefiles and LIDAR data were then projected to match the projection of the USGS image (UTM zones or stateplanes). The original datum and projection of each file at the time of download is included in the metadata.

Descriptive Image Statistics

The following information provides a summary of the data that demonstrates a balance between urban and suburban regions and shows the quantity and area of the buildings that can be expected in each image.

Buildings by Image

The bar graph below shows the number of buildings for each image labeled in the same format as in the files. An orange bar symbolizes that we have labeled this image as belonging to a suburban area, whereas a gray bar symbolizes that we have labeled this image as belonging to an urban area.

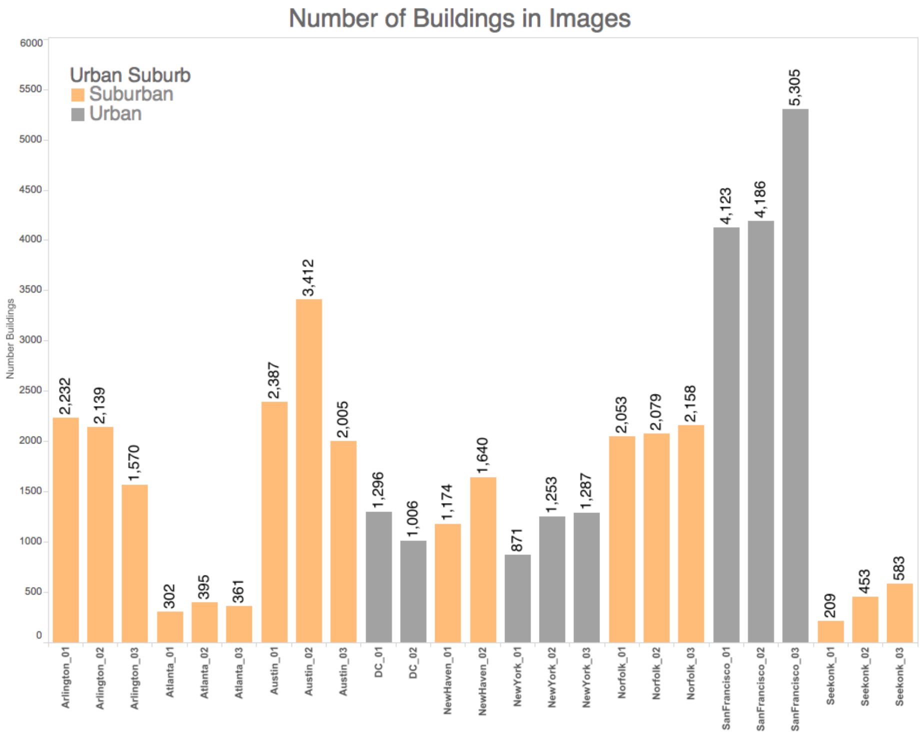


Figure 4 - Shows each image, whether it is urban or suburban, and the number of buildings in the image

The histogram below shows the number of buildings by building footprint area. The subplot on top is for suburban buildings whereas the subplot on bottom is for urban buildings.

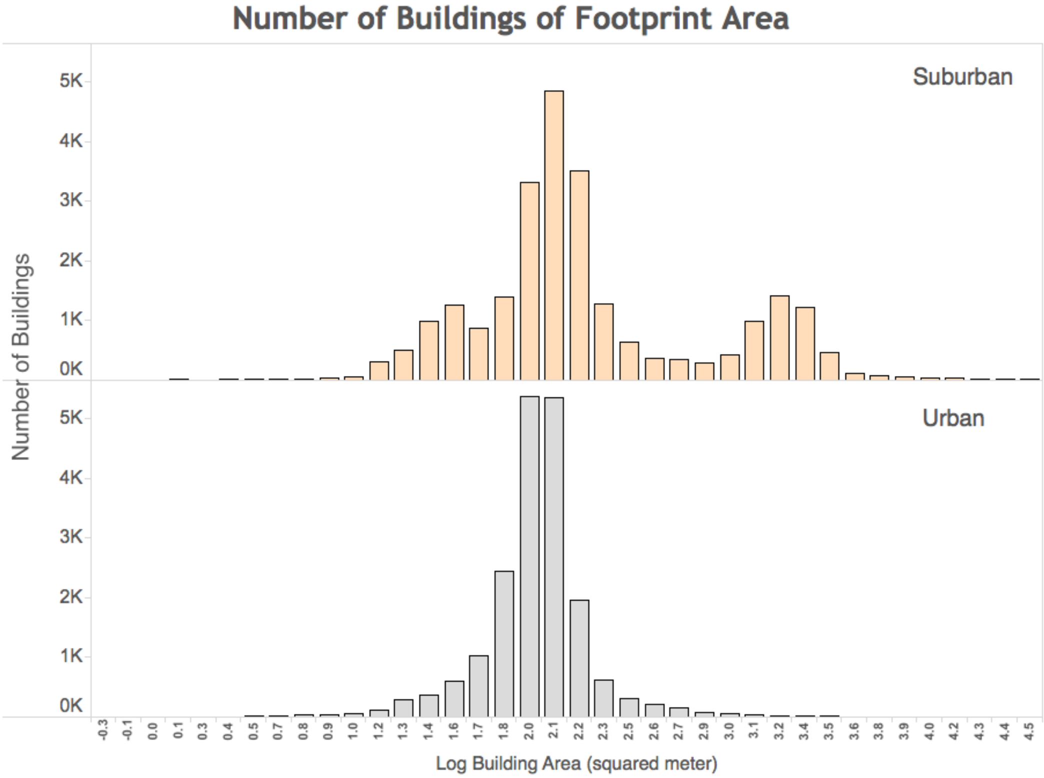


Figure 5 - Histograms of building area on a log scale for urban and suburban areas

The bar graph shows the building density for individual cities, labeled suburban or urban. We define building density as the area comprising buildings over the total area in an image. The box and whisker plot shows building density of individual images, with a box for suburban and another for urban. This plot, as would be expected, shows that the urban images have higher building density than do suburban images.

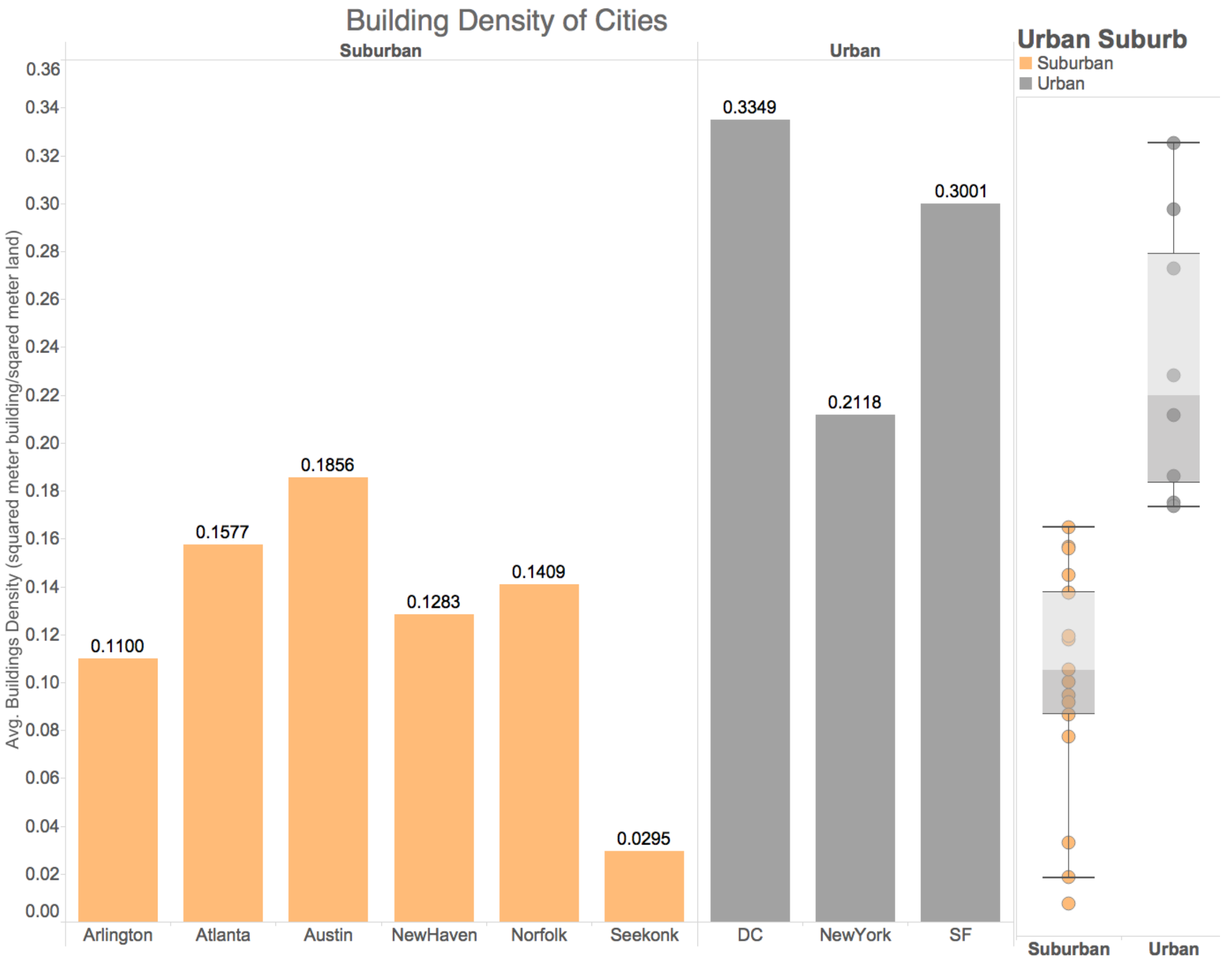


Figure 6 - The bar chart shows the average building densities of the images for each city. The box plot shows the densities for each image.

The four circle charts convey total building area, average building size, average building density, and number of buildings for suburban city images and urban city images.

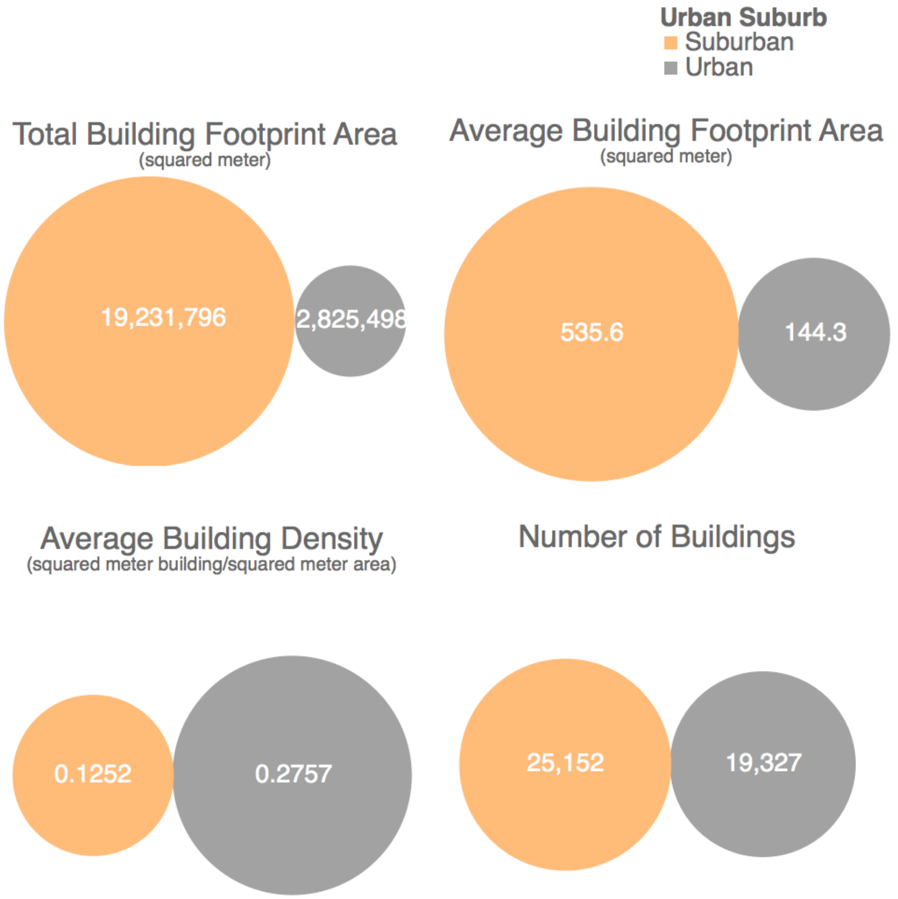


Figure 7 (clockwise from top left) - Total building area shows the total area covered by buildings in our urban and suburban images. Average building size shows the average area of buildings in urban and suburban areas. Number of buildings shows the total buildings in urban and suburban areas in our images. Average building density shows what percentage of the image is covered by buildings.

Methodology

This section describes each of the steps for creating this ground-truth dataset. This processed included (1) selecting candidate regions with all available data, (2) cropping all data to a common geographic extent, (3) converting and interpolating LIDAR from point-cloud binary data to uniformly-sampled raster data, and (4) providing building and road shapes indexed to pixel coordinates for each image. Figure 8 shows the sequence of processing steps undertaken and this sections describes each step in detail and for future reproducibility.

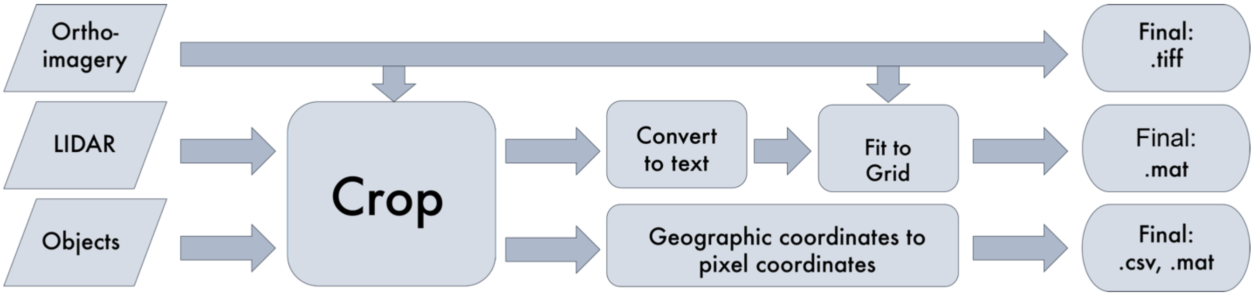


Figure 8 - Overview of processing steps

Area Selection

1. Go to USGS EarthExplorer and scan the overlapping area where LIDAR data and orthoimagery both exist.
2. Choose cities in diverse locations in the overlapping area and roughly balance the number of buildings in urban and suburban areas.
3. Go to Open Street Map website and check whether building annotations in selected cities are complete or not. Delete cities in which the annotations are not complete.
4. Choose orthoimageries in selected cities. The orthoimageries should have enough buildings and low tree densities.

Orthoimagery

Download

1. Login to USGS EarthExplorer (http://earthexplorer.usgs.gov/)
2. Select desired area on the map
3. Enter other parameters
    a. Date range: Isolate most recent data
    b. Resolution: .3m (1ft) or better
4. Under the "Data Sets" tab, click the box next to "High Resolution Orthoimagery" under "Aerial Imagery."
5. Export metadata as CSV

6. Add images to bulk download – for each page of results, click “Add All Results From Current Page to Bulk Download.”
7. Use the Bulk Download application to download the images to the desired location

Building Polygons and Roads

Download

Object locations are from the open source mapping project Open Street Map. Although it is possible to export data directly from Open Street Map, there are websites which provide the data packaged nicely as shapefiles. Two such websites in which Open Street Map data is available are Geofabrik and Trimble Data Marketplace. Geofabrik provides the data by state, while Trimble only provides data for certain metro areas. However, Geofabrik's data doesn't include multipolygons. Multipolygons are most commonly used for buildings that have a "hole" in the middle, like an interior courtyard.

Geofabrik

1. Go to the Geofabrik download site (http://download.geofabrik.de)
2. Using the list on the page, navigate to the sub-region you would like to download data from.
3. Once you've found your desired area, download the data in .shp.zip format

Trimble Data Marketplace

1. To download Open Street Map from Trimble, first go to its website http://market.trimbledata.com/.
2. On the right, there is a searchbar labeled "Location." Use this to search for the city in which you want objects data from.
3. Under the searchbar on the left, click on "Streets" to narrow your search. Hover over Provider and choose OpenStreetMap.
4. Options starting with "OpenStreetMap:" and an image of the area that it covers will appear below the searchbar. When you put your cursor on each one, the map will be shaded for the area in which the shapefile covers. Find the result that covers the relevant area for your orthoimagery.
5. When you have found the shapefile you would like, click on it. Define parameters before ordering:
    a. Region: if you want to specify a certain part of the shapefile, draw it on the map.
    b. Layers: select the layers of the shapefile you want for the project. There are many object choices. For this project, "Buildings" was selected.
    c. Datum/Projection: Select World, NAD83 projection.
    d. File Format: Keep the shapefile as an ESRI shapefile.

After you have done so, order the data and add to cart. You can expect an email from Trimble and then you can download the data.

ArcGIS Processing

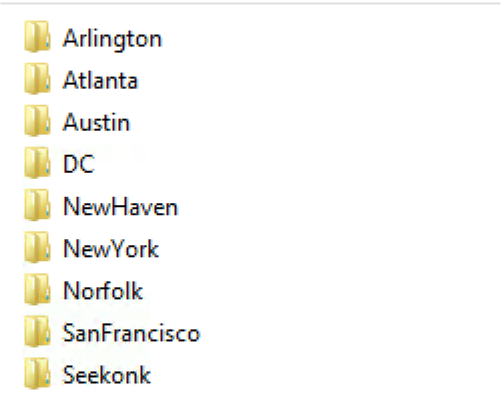
The Open Street Map shapefiles (both the building footprints and the roads) need to be cropped to the same extent as the USGS orthoimagery. To make this process less time consuming, it is recommended you build a simple looping model in ArcGIS to bulk-crop the shapefiles.

1. Use the Repair tool on the downloaded shapefiles. This fixes geometric anomalies and removes potentially corrupted or NAN features included in the shapefiles. These may otherwise cause errors in later steps.
2. Create a polygon shapefile mask for the USGS image. Do this using the Raster Calculator and setting all values in the image to 0. Then use the Raster to Polygon tool to create a single rectangular polygon that has the same extent as the USGS image.
3. Use the Intersect tool to crop the OSM shapefile to the newly created polygon mask.
4. For the building footprints only, use Add Geometry Attributes to create add the centroid latitude and longitude (we use NAD83 geographic coordinates) to the data. Also add the shape area in meters squared.

After completing these steps, you should have a building footprint or road shapefile with the same extent as the USGS image. Note that edge features might have been truncated and do not necessarily represent a whole building.

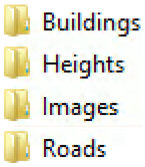
MATLAB Processing

For the provided Matlab code, the files should be downloaded and arranged in a directory that as a subdirectory for each city:



Arlington
Atlanta
Austin
DC
NewHaven
NewYork
Norfolk
SanFrancisco
Seekonk

Then should be subdirectories in each city folder for Buildings (building shapefiles), Heights (Lidar text files), Images (Orthoimagery), and Roads (road shapefiles).


Buildings
Heights
Images
Roads

Buildings
Matlab code is available in github in the following link:
(https://github.com/energydatalab/building-dataset)
Download the following codes: runbuildings.m, cell2csv.m, coord2pix.m, processUSGS.m in the same folder. Then run runbuildings.m.

Roads
Matlab code is available in github in the following link:
(https://github.com/energydatalab/building-dataset)
Download the following codes: runroad.m, cell2csv.m, coord2pix.m, processUSGS.m in the same folder. Then run runbuildings.m.

LIDAR Height

Download

NOAA

1. Go to the NOAA website (https://coast.noaa.gov/digitalcoast/) and choose Elevation
    a. The orange area represents where LIDAR data is available
2. Select desired area
    a. Click ‘Draw Search Area’
    b. If you want to delete the area you just chose or want to redraw the area, you can click ‘Reset Map’
3. You will see the data available on the right side. Click the cart button on the data set you want to download. If you cannot click on the button, make the selected area smaller because there is a limitation on the output.
    a. If you want to download the whole data set, which is represented by the blue area that appears when you move your mouse on the item, you can click ‘Bulk Download’ instead of the cart icon.
4. Download the data
    a. After you click ‘Add to Cart’, click the ‘cart’ button on the right upper corner.

USGS

1. Login to USGS EarthExplorer (http://earthexplorer.usgs.gov/)
2. There are two options to find the area to download LIDAR for:

    a. In the map on the right, zoom in on the approximate location of the USGS images. Click on the map the vertices of the area which contains the USGS images.
    b. Use the "Add Coordinates" under the "Coordinates" section to manually enter the coordinates of the vertices of the area containing the USGS images.
3. Enter date parameter in the "Date Range" section to isolate updated data.
4. Under the "Data Sets" tab, click the box next to "LIDAR" under "Digital Elevation.
5. Export metadata as CSV.
6. Add images to bulk download – for each page of results, click “Add All Results From Current Page to Bulk Download.”
7. Use the Bulk Download application to download the images to the desired location.

ArcGIS Processing

We use rapidlasso's LAStools toolbox (https://rapidlasso.com/) to crop LIDAR data. Their ArcGIS toolbox allows for convenient and easy processing in ArcMAPS. The LAStools Production toolbox allows for bulk processing of LIDAR files.

1. Download the LIDAR data from NOAA, USGS, or another source. Keep note of whether the downloaded data has a .laz or .las extension.
2. Open ArcMap. There will be a window that pops up. Click to work on a Blank Document and press OK.
3. Next we will need to add the necessary toolboxes to process LIDAR data in ArcMaps. To the bottom left, there will be a section for Arc Toolbox with a red icon. Right click the icon, and press the Add Toolbox... which should be the first of the choices. Navigate to your LASTOOLS folder. Then press the ArcGIS_toolbox. Open LAStools.tbx. It should be the first choice with a red icon. Repeat this step to also add LAStools Production.tbx.
4. You will see that the new toolboxes have been added on the right. The next step depends on the type of LIDAR data you have downloaded. LIDAR data is in tiles. Try to overlay your desired image onto your LIDAR data. Most likely, your LIDAR data will be in tiles, and the USGS image covers more than one LIDAR data file. If this is the case, then go to A, but if not then do B.
    a. Double-click on LAStools Production. From the down bar, double-click lasclipPro. For the input folder, browse and navigate to the folder in which your downloaded LIDAR data is. If you installed it with the .laz extension, set the input wildcard(s) to *.laz. If you downloaded it with the .las extension, set it to *.las. The box next to "merge input files on-the-fly into one " should have a check in it. Click the box if it does not. For the clip polygon, choose the shapefile (.SHP extension) which is a polygon of the desired area. (This is the same shapefile that was used to crop the

building shapefile data.) Make the "output format" las. For the output file, navigate to the folder in which you want to put the resulting las file, and for the File name, include a .las extension. For example, it may be named 'cropped_lidar.las'. Adding the extension to the output file name is important. Click ok.
  b. Instead of click on LAStools Production, double-click LAStools. Choose the lasclip from the down bar. Do the same as A), except for navigate to the specific .las file which is your downloaded data that includes the desired area. Skip the directions about the input wildcard and the merging files.

It will take some time to complete the clipping of the LIDAR data. When it is done, you have your cropped LAS file! The next step is only necessary if you want to visualise it.

Visualizing the Cropped LIDAR

1. Open up ArcCatalog. In the tab "Contents", navigate to inside the directory in which you want to save your file. Right-click or press File on the upper left corner. There will be a downbar with a choice of 'New'. Hover over it and you will see an option of 'LAS Dataset'. Name the dataset. Double-click the new dataset you have created. Under the second tab, 'LAS Files', click 'Add Files...' And navigate to the new clipped LAS file. *This is not the file you initially installed, it is the one created from Step 4.
2. Go back to ARCMap. In the Catalog on the right, navigate to where you saved your LAS Dataset (with .lasd extension) that you created in Step 5. Drag onto the main screen to view it. (It will take time to load).

MATLAB Processing

For our project, the use of multiple types of data presents a few complications. One of these complications stems from the different ways LIDAR data and high-resolution orthoimagery data are stored. While orthoimagery is divided into uniform pixels in a .tif file, LIDAR is stored as discrete points in a .las file. We need the LIDAR data to correspond to the orthoimagery; that is, we need the discrete LIDAR points to be assigned to a grid of “pixels” the same size as the orthoimage. This document details the procedure we used to convert the .las data to a raster, using a 5000x5000 USGS orthoimage as a reference.

Convert LAS to text

The first step is to convert the raw .las file into a text file readable by Matlab. We used the las2txt tool from the LAStools suite to do this. For detailed information about LAStools and las2txt, check the README files included with the software. The easiest way to use las2txt is to run the program from the command line.

1. In the command line, navigate to the directory where your .las files are located.
2. Run the command with the following format:

   ```
   path_to_LAStools\bin\las2txt –i “lidar.las” –odir “output_directory” –parse xyz
   ```

   The text file is written to the given directory with the same name as the .las file, but with a .txt extension (e.g., lidar.las turns into lidar.txt). The text file consists of three columns, indicating x-coordinate, y-coordinate, and height.

   This command is easily applied to multiple files, and works for both .las and .laz files. The following command will turn all .laz files in the working directory to .txt files in the output directory.

   ```
   path_to_LAStools\bin\las2txt –i “*.laz” –odir “output_directory” –parse xyz
   ```

Create rastered data

Now we will use the LIDAR text file in Matlab to create a gridded data set.

1. Load the text file into Matlab. This is easily accomplished by using the `load` function:

   ```
   lidar = load("lidar.txt");
   ```

   This will create a matrix with three columns: x, y, and height, with the number of rows corresponding to the number of points in the .las file.

2. Load the image into Matlab. Since the LIDAR grid needs to correspond exactly to this image, we will use the image to create the grid. Luckily, Matlab has a built-in function to handle GeoTiff images:

   ```
   [img, data] = geotiffread("picture.tif");
   ```

   This creates two variables – `img`, which holds the RGB values for the picture, and `data`, which holds geospatial reference data. If you don’t need to use the image, `img` can be replaced with a ~ to save memory.

3. Separate x, y, and height values, find x and y ranges, and find the grid size.

   ```
   x = lidar(:,1);
   y = lidar(:,2);
   height = lidar(:,3);
   ```

```
xLimits = data.XWorldLimits;
yLimits = data.YWorldLimits;

gridSize = data.RasterSize;
```

`xLimits` and `yLimits` store the extreme coordinates occupied by the image. `gridSize` stores the number of rows and columns in the image.

4. Create the grid. First, we must create the x and y values that form the grid.

```
xRange = linspace(xLimits(1), xLimits(2), gridSize(1));
yRange = linspace(yLimits(1), yLimits(2), gridSize(2));
```

Next, combine these values into the grid.

```
[XI, YI] = meshgrid(xRange, yRange);
```

5. Fit the LIDAR data to the grid. Since LIDAR data isn't uniform, there may be spots on the grid where no data is recorded. To avoid this, we use interpolation to estimate the missing values. The Matlab function `griddata` includes an argument for interpolation type, but is limited to five options. After trying these methods, we determined that the nearest-neighbor method worked the best for our purposes. The following code creates a rasterized height data set.

```
ZI = griddata(x, y, z, XI, YI, ‘nearest’);
```

The rasterized data set `ZI` is ready for use.