Parking Provision Project:
Counting Off-Street Residential Parking in the UK

1. Introduction

1.1 Background and motivation

The Office for Low Emission Vehicles (OLEV) with the Environment Statistics (ES) team at the Department for Transport (DfT) are currently planning the future provision across the UK of electric vehicle (EV) chargepoints, as part of the commitment to reducing emissions by 2050. We imagine a future where all vehicles are fully or partially electric, and every vehicle requires adequate access to a chargepoint. Charging an EV can take considerably longer than refuelling a combustion engine. Different chargepoint types (slow, fast, rapid) can charge an EV partly or fully at various speeds, depending on power rating, ranging from 5 minutes to 10 hours. The demand on the electricity grid is an important consideration, and to mitigate this the majority of charging should take place at night, outside peak hours of electricity consumption (OLEV, 2011). Thus, it is anticipated that there will be many chargepoints located on, or nearby the property of the EV owner. Hence it is important to understand parking availability. It is expected that properties will fall into one of the following three categories,

1. Off-street parking: the property has a driveway or garage.
2. Adequate on-street parking: there is enough room for all or most vehicles to park within some specified short distance of the property.
3. Inadequate on-street parking: there is **not** enough room for all or most vehicles to park within some specified short distance of the property.

There is no definitive existing dataset that quantifies residential parking availability for the UK. There are various datasets which may include some information. For example, HM Land registry and estate agency collections, such as Zoopla, may contain garage or off-street parking details for a property.

1.2 Aims and Objectives

In this project I will restrict my aim to quantifying the proportion of uncovered off-street parking in a residential 1 km^2 test area in the borough of Ealing. More specifically I aim to examine the front garden area of residential properties to determine if they may be used for parking and hence may accommodate a private EV chargepoint powered via the household electricity supply. At a later stage, beyond the scope of this assignment, I expect to scale the method to obtain UK wide figures at the local authority (LA) and lower super output area (LSOA) level. I intend to use manual labelling of parking within the test area to validate the outputs of this project. Various high-level data and rougher estimates of off-street parking also exist to provide context.

1.4 Ethical issues

No personal data is used. The aerial images are pre-processed to remove people and the angle and resolution is insufficient to identify number plates of vehicles. One could extract the address of a property in the image data, then, for example, make inferences on the attributes of the owner based on the size of the property and other features within the property extent. However, there is no direct link in these data from images and addresses to names or other personal attributes.

2. Related work

2.1 Emu Analytics

Most notably Emu Analytics developed a product using Land Registry INSPIRE Index (LRII) polygons, which show the extent of a property, and single structure outlines from Ordnance Survey Open Map (OSOM). The distance from the property frontage to the building is used to calculate a likelihood that the property can support off-street parking (Emu Analytics, 2018). As a baseline of

potential parking this product has merit, however, they are not able to restrict the data entirely to residential properties, nor delineate all connected properties, such as terraces. More pertinently it is not free, and as a public sector enterprise, DfT has access to superior, and free, data sources.

2.2 ONS Green spaces

The Office for National Statistics (ONS) recently conducted a study into the proportion of vegetation in residential gardens (Bonham, 2019). This work is particularly useful as my work is interested in the precise opposite: that is identifying the non-vegetation, or manmade surfaces in residential gardens. ONS reviewed several different methods of segmenting an aerial or satellite image into areas of vegetation. The main problems occur when classifying shaded areas, and areas that undergo dramatic seasonal change where vegetation may be covered by snow in winter or parched yellow in the summer. Their study recommended two approaches to me, the Normalised Difference Vegetation Index (NDVI) and the Hue, Saturation Value (HSV) colour palette, which I will detail in the next section.

2.2 Remote sensing

The NDVI value is calculated for each pixel in a Colour Infra-Red (CIR) image. While a Red, Green, Blue (RGB) image contains only three channels of the visual spectrum, a CIR image contains a fourth channel, of Near Infra-Red (NIR) values, or it may drop the Blue channel and contain only NIR, Red and Green. Each channel contains a value in the integer range [0, 255] representing the intensity of those light frequency ranges. NIR frequencies are not visible to the human eye. They comprise frequencies of light that healthy vegetation cannot use in photosynthesis, and therefore most reflect. Hence the NIR intensity values for vegetation tend to stand out against those for the other channels. The NDVI value is a ratio of the difference in NIR and Red values over their sum, thus it falls in the interval [-1, 1] with positive values representing vegetation.

$$NDVI = \frac{NIR - Red}{NIR + Red}$$

This index is widely used, though it has limitations as noted above. There exist many variations on this index, as studies often seek to distinguish between different types of vegetation (Xue & Su, 2017).

The HSV colour palette represents colour in cylindrical coordinates where Hue, Saturation and Value are distances in the azimuthal, radial, and height dimensions, respectively. Instead of representing the final colour by combining the values of red, green and blue, as in the RGB system, in the HSV system only hue specifies the colour frequencies, while saturation and value modify the amount of grey and brightness (Jacob Rus, 2010). Hence, a single channel, hue, can be used to specify a colour range to segment an image.

2.3 Minkowskii difference

The Minkowskii *sum* of two sets of points, A and B is the set of points obtained by the vector addition of every pair of points in each set, for example

$$A = \{(1,2),(3,4)\}, \qquad B = \{(0,1)\} \implies A \oplus B = \{(1,3),(3,5)\}$$

The Minkowskii difference, $A \ominus B$, is not quite so straightforward. $A \ominus B$ is comprised of points c such that the vector sum of each c with points in set B is a subset of A,

$$A \ominus B = \{c \mid (c \oplus B) \subseteq A\}$$

By extension to a set of points defining a polygon what this means practically is that if $A \ominus B$ is non-empty then polygon B fits inside polygon A, as demonstrated in *From Minkowskii Sum to Concave Hull* (Zhou & Simmons, 2019). I considered using this approach for determining if a

parking space sized rectangle would fit within a residential front garden as an alternative to finding the spatial intersection.

3. Methodology

3.1 Tools

All the code is written in python 3 in Jupyter Notebooks, I also used MS Excel to prepare a form for manual labelling of images and QGIS to crop the larger geopackage files to the test area before processing. The main python libraries I used for geospatial and image analysis are GeoPandas, Shapely, Rasterio, OpenCV (cv2) and dependents.

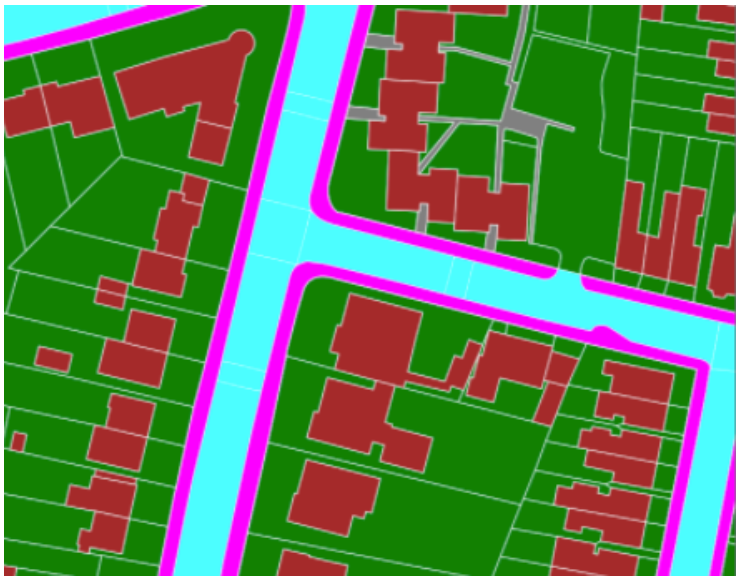
3.2 Data

The methodology developed is very much shaped by the data available. The main data used in this project includes aerial photography raster data and vectorised polygon data of the UK. These geospatial data are accessible to public sector employees via the Public Sector Geospatial Agreement (PSGA).

Ordnance Survey Master Maps (OSMM):
Topography Area (TA) Layer and AddressBase Plus (AB+)

The Ordnance Survey Master Maps series comprises several layers of maps of the UK. They use the British National Grid (BNG) coordinate reference system (CRS). The Topography Area layer contains vectorised polygons of every shape that together make up a jigsaw of the UK, for example gardens, roads, pavements, buildings, see Figure 3.2.1. Polygons only intersect on adjacent edges, and do not overlap. Other useful fields include the polygon unique identifier, or topographic identifier (TOID), and various descriptive fields which enable detailed filtering (Ordnance Survey, 2016a). AddressBase Plus contains a unique property reference number (UPRN) for every postal address in the UK, along with point geometry for its location and the building TOID that it matches in the TA layer. Each address has an address type, for example residential dwelling, commercial etc. The classification scheme contains over 500 address types organised in a hierarchy, so it is possible to filter at different levels of detail (Ordnance Survey, 2018).

Figure 3.2.1: OSMM TA layer, buildings (brown), gardens (green), paths (grey), pavement (magenta), road (cyan).



Approximate scale: ↔ 3 m

HM Land Registry INSPIRE Index (LRII) polygons

The Land Registry INSPIRE Index polygons contain polygons of the full extent of a property. These polygons are based on OSMM data and align very closely, but not always perfectly as they indicate the location of the registered land that comprises the property extent (HM Land Registry, 2020). Figure 3.2.2 shows an example of the TA layer and LRII polygons overlaid, with some examples of TA polygons split by the LRII polygons highlighted in yellow.

Figure 3.2.2: LRII polygons (bold black outline), TA polygons (grey outline) and misalignment (yellow) examples.

Approximate scale: ↔ 2 m

Aerial Photography Great Britain (APGB) RGB 25 cm and CIR 50 cm

APGB products are raster images of 1 km^2 grids that align with the BNG. They combine multiple aerial photographs taken between 1st April and 31st October over several years. The RGB 25 cm renders 1 pixel length for every 25 cm of ground level distance and contains values in the range [0, 255] for three channels of the visible spectrum, Red, Green and Blue (APGB, 2020a). The CIR 50 cm resolution images contain the Near Infra-Red (NIR) channel as well as the visible red and green channels (APGB, 2020b).

Figure 3.2.3: RGB 25 cm example

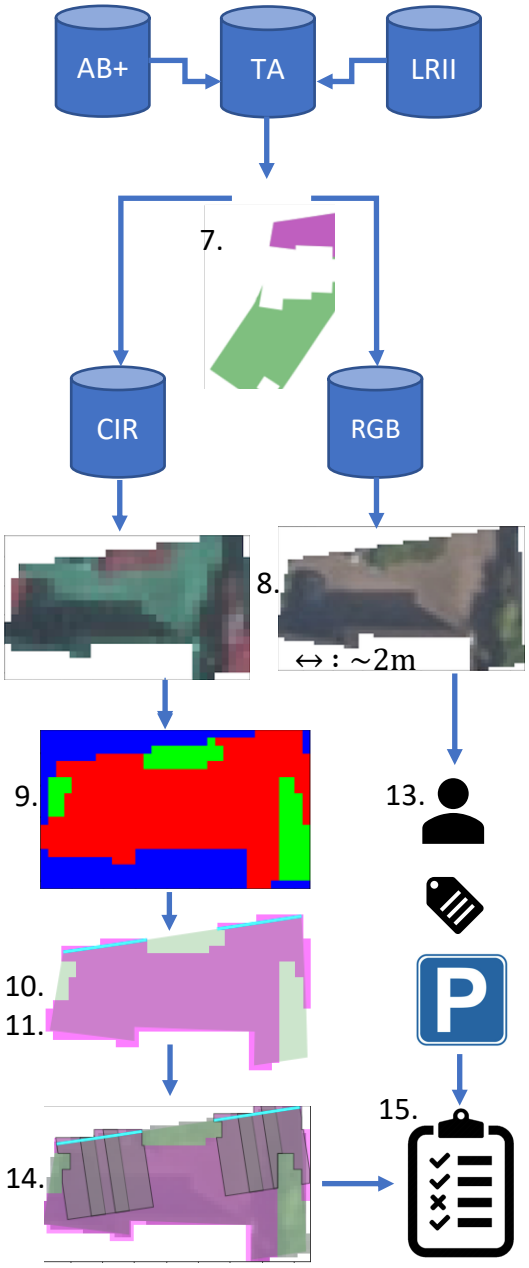
Figure 3.2.4: CIR 50 cm example

3.3 Methodology steps and diagram

The method I developed assigns a label of parking or no parking to a residential property by isolating the manmade surface area in the 'front garden' of a property and checking if this area has both sufficient roadside access and is of sufficient size and shape to accommodate a vehicle. Throughout I refer to properties and TOIDs (the unique garden polygon identifier), interchangeably.

Below are the main steps of the methodology, with the script code in brackets (see appendices). Figure 3.3.1 shows the effects of steps 7 to 15 on an individual property. The first image in the flowchart shows the full extent of the garden in green and magenta (7), the roadside is on the north edge and the white cut outs are the building footprint and a shed in the rear garden. The magenta area is the section of the garden that is within 9 m of the roadside edge of the property. These sections make up the **cropped garden** polygons that are used to jigsaw the RGB and CIR images (8). The CIR image is segmented by NDVI value per pixel into manmade surface (red), vegetation (green) and non-garden (blue) (9). The manmade surface raster is vectorised (pink) and for each roadside linestring (cyan) we fit parking sized rectangles, incident with the roadside edge and perpendicular to it, in three different positions (10, 11, 14). The RGB image is used for manual labelling and validation (13, 15) against the parking rectangle fitting algorithm.

Figure 3.3.1



1. Establish a test area
2. Get residential dwellings from AB+ and corresponding TA building TOIDs (ppp06).
3. Filter TA for private gardens and roadside polygons respectively (ppp06).
4. Use spatial join to select only gardens that intersect with a residential building and a roadside (ppp06).
5. Use LRII polygons to link each RD building to its roadside garden within the property extent, using representative points (ppp06).
6. For each garden collect the roadside access edge which is the linestring intersection between the garden and roadside polygons (ppp10).
7. Cut off connected rear gardens by taking the intersection between each roadside garden and a 9 m buffer from the roadside linestring, to create the cropped gardens (ppp10).
8. Write jigsaw function to cut out each cropped garden polygon from the CIR and RGB raster images (ppp11).
9. For each jigsawed CIR image calculate the NDVI value for each pixel, from the NIR and Red channels, and segment the image into areas of manmade surface (red) and vegetation (green) (ppp13).
10. Vectorise the manmade surface areas, fix any invalid polygons and fill any holes less than 1 m^2 (ppp13).
11. Intersect the roadside linestrings with the vectorised manmade surfaces to establish access edge to the manmade surface area (ppp13).
12. Filter out areas that cannot accommodate parking: areas less than 12.5 m^2, those with no roadside, or roadside linestring less than 2.4 m.
13. Manual labelling of RGB raster images into parking or no parking (ppp14).
14. Fit three parking rectangles algorithm, establish a cut off threshold (ppp15).
15. Validate algorithm against manual labelling (ppp16).

3.4 Establishing a test area

Initially I chose two test areas, Ealing Borough and Cornwall, as representative of suburban and rural areas. However, the vector and raster data for these areas is still very large, in the GB range and not practical for my computer and its modest 8GB of RAM. The image data is organised into 1 km square grids, aligned to the BNG reference system, so it made sense to pick one of these. This also naturally suggests a method of scaling the process in future. I chose grid TQ1980, which is a subdivision of the 100 km by 100 km TQ grid, beginning 19 km east and 80 km north of its south west corner. The TQ grid itself begins 500 km east and 100 km north of the BNG datum (Ordnance Survey, 2016b). TQ1980 contains my property so it had the advantage of being a familiar area with the option of street level visual inspection if required. I used QGIS to manually crop the AB+ and TA layer vector data to the TQ1980 area. I did not manually crop the LRII polygons as I had these for Ealing and since they contain far less detail that the equivalent area TA layer the size is manageable at about 80MB, so I could crop it within the script.

Figure 3.4.1: TQ grid, 100×100 km

Figure 3.4.2: TQ1980 test area, 1×1 km

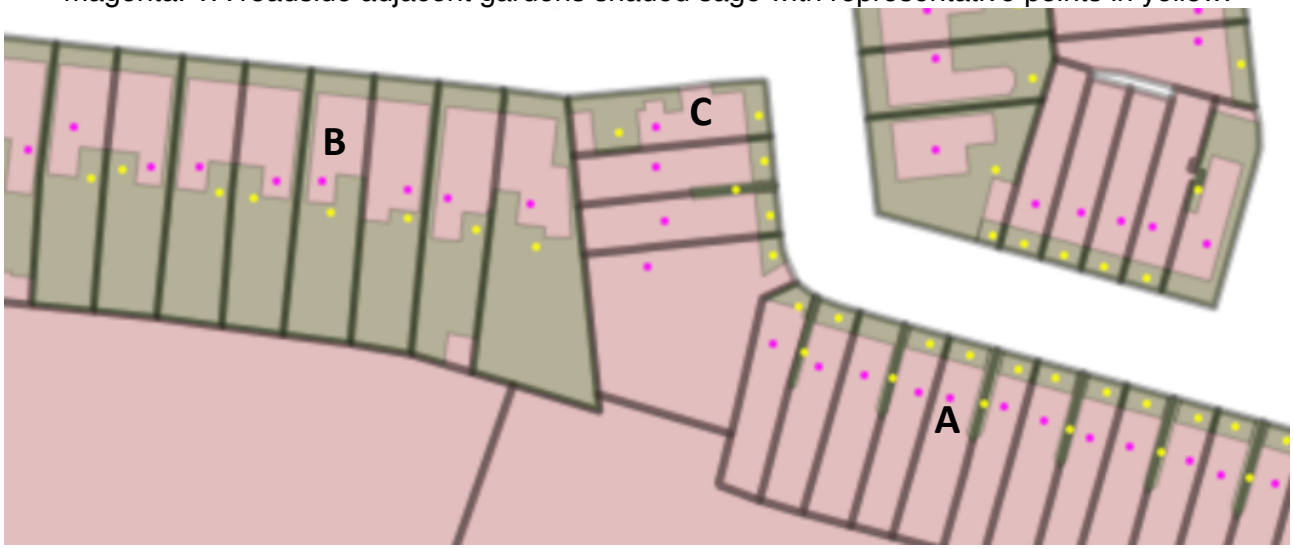
3.5 Data filtering and linking

To extract just the front gardens of residential dwellings requires linking between the three datasets AB+, TA and LRII, and filtering via spatial joins for gardens that are adjacent to the roadside.

The AB+ classification field contains values with the code prefix ‘RD’ for residential dwelling. AB+ also contains point geometry of the building location and the TOID corresponding to the building polygon in the TA data. The TA data can be filtered for private gardens, with no distinction between front and rear gardens, and for roadside objects such as pavement and street furniture. Using a spatial join for intersection of gardens with roadside polygons ensures the removal of most of the unconnected rear gardens. Next, using representative points for the buildings and gardens we use a spatial join to add the LRII polygon identifier, establishing a link between the building and garden contained within each property extent. A representative point is a computationally cheap way to create a point that is guaranteed to be inside a polygon.

Figure 3.5.1 shows the resulting roadside garden polygons. Where a building completely disconnects front and rear gardens the rear gardens are removed, as in A, which is part of a row of terraced houses. Where a property is semi or fully detached the front and rear gardens are one polygon, as in B. Corner properties, such as C, are problematic as they have roadside adjacency on at least two sides, and so a rear garden that is disconnected will be retained.

Figure 3.5.1: LRII property extent with bold black outline. Residential building representative points in magenta. TA roadside adjacent gardens shaded sage with representative points in yellow.



Approximate scale: ↔ 3 m

3.5 Cropped gardens

Retaining rear garden areas is undesirable as there is usually no vehicular access from the road, yet there may be sufficient manmade surface areas, such as a patio that would appear large enough for parking. Although these could have been filtered out at a later stage it is useful to reduce the data size as much as possible before image processing.

To remove these areas I collected the roadside linestring for each property, by intersecting the front garden polygons with the roadside polygons (pavements), then buffered this linestring to 9 m, and overlaid it with the garden polygons again. Figure 3.5.1 shows the resulting cropped garden areas in magenta, created from an intersection of the original garden with its buffered roadside linestring edge (cyan). The choice of 9 m was somewhat arbitrary, it needed to be large enough to ensure capture of the entire front garden area, but not too large to spill into the rear garden space. Rear gardens of corner properties were still retained.

Figure 3.5.1: Cropped garden polygons (magenta), original garden outline (green), buffered roadside linestring segments (cyan), building polygons (beige)

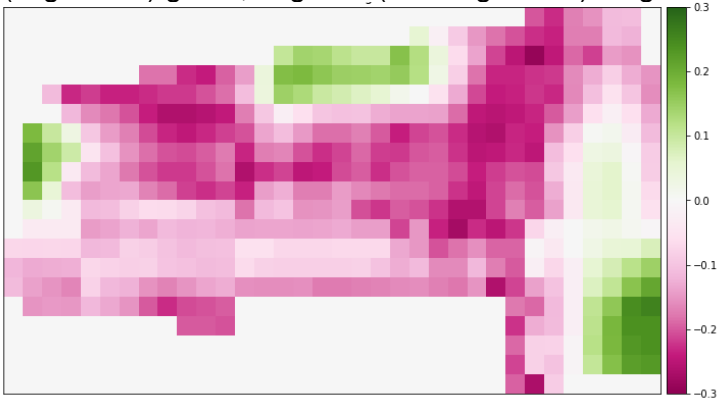
Approximate scale: ↔ 8 m

3.6 Image jigsaw and segmentation

I wrote a function called jigsaw that took a cropped garden polygon as input and used it as a mask over the test area raster image, creating a new raster file for each, and preserving the geospatial metadata. This was applied to the CIR and RGB APGB data. For each cropped garden CIR image I calculated the NDVI value per pixel and used this to segment the image. The jigsaw process clipped the image to its rectangular bounds, and any pixels outside the cropped garden polygon were set to white, that is 255 in each of the three channels. This resulted in a huge number of zero NDVI values, which occur when the NIR and Red channel value are equal. Note that Zero NDVI values also occur naturally in the data, however very infrequently. The 1000 images used in Figure 3.6.2 together comprise nearly 300,000 pixels and of these only 1.2% were naturally occurring zero NDVI values.

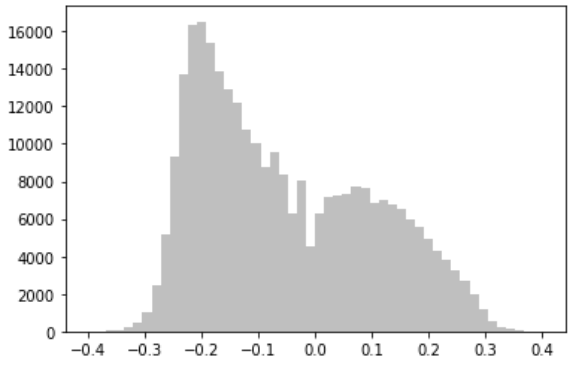
NDVI values lie in the interval $[-1, 1]$, however, as can be seen from the histogram in Figure 3.6.2 the vast majority lie in the smaller interval $[-0.4, 0.4]$. Figure 3.6.1 shows a heatmap of the NDVI values for an example garden, the colour scale is adjusted to fit the range for this particular image; $[-0.3, 0.3]$.

Figure 3.6.1: NDVI value heatmap example, positive (vegetation) green, negative (non-vegetation) magenta



Scale: 1 pixel $\equiv 0.5\ \text{m} \times 0.5\ \text{m}$

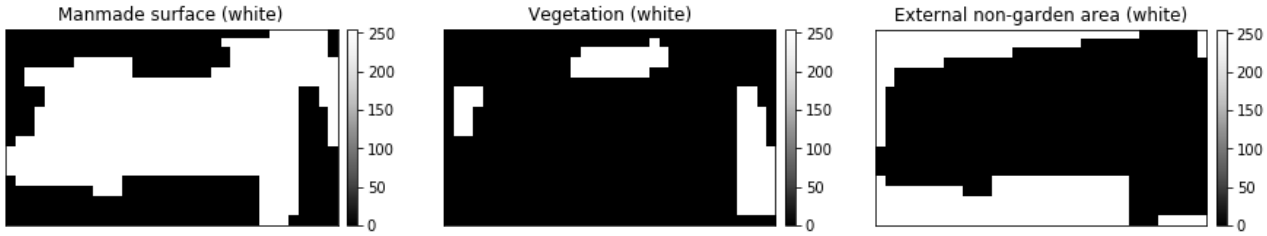
Figure 3.6.2: Distribution of NDVI values $(\neq 0)$ for 1000 cropped CIR images.



NDVI value (zeros removed)

Coincidentally there are three areas to segment and three channels. Hence, it is logical to segment each image by mapping each area to full intensity, 255, in one channel only. Thus, the external garden area, (the white pixels occurring as an artefact of the jigsaw procedure) are mapped to full blue. Areas with positive NDVI values, and any remaining natural zeros, are mapped to full green and areas with non-positive NDVI values mapped to full red. These arrays are combined to create the segmented raster images. For each pixel precisely one channel has value 255 and the other channels are zero, as shown in Figure 3.6.3.

Figure 3.6.3: Red, Green, Blue channels of a segmented image shown in monochrome
Scale: 1 pixel $\equiv 0.5\ \text{m} \times 0.5\ \text{m}$

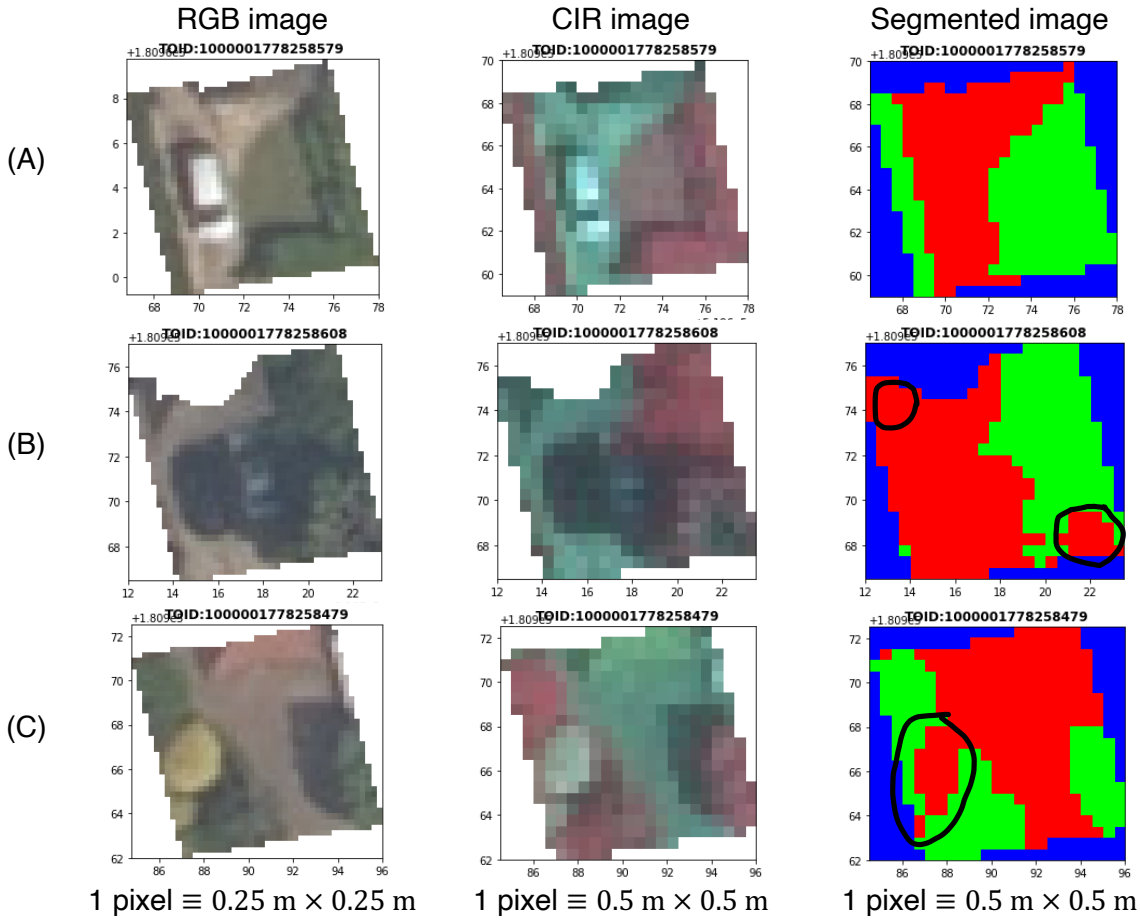


Red channel Green channel Blue channel

Some further examples of the results of the jigsaw function and segmentation can be seen in Figure 3.6.4. Example (A) shows clear delineation between the driveway and the lawn and

shrubbery. Example (B) has some debateable areas classed as manmade surface (red) in the top left and bottom right, ringed in black on the segmented image. Comparing to the RGB image these look more likely to be lawn and foliage, respectively. Similarly, example (C) shows a yellowish area, which may be parched lawn, classed as manmade surface. Areas in shadow in these examples appear to be mostly segmented correctly.

Figure 3.6.4: Three examples, (A), (B), (C), of cropped gardens jigsawed from RGB and CIR raster images and then segmented by NDVI value.



3.7 Vectorization and access

This is the final step in preparing the polygons for the rectangle fitting algorithm. The segmented raster images are loaded and the manmade surface area (red) channel array is converted to a vector polygon and appended to a geodataframe, preserving the TOID reference and CRS. This dataframe comprises the final set of 2626 Vectorized Manmade Surface (VMS) polygons for 1329 residential garden TOIDs (TOIDs may have multiple VMS polygons) that is checked for parking potential.

Roadside access is a key factor, hence for each TOID in this data set we also collect the linestring geometry representing the intersection of the VMS polygon with the roadside. For each TOID we now have one or more polygons representing manmade surface areas on the property and a separate geodataframe of roadside access linestrings per TOID.

Figures 3.7.1 and 3.7.2 below show examples of some VMS polygons with one or more intersecting roadside access linestrings. This is elaborated in section 4.

Figure 3.7.1: RGB image with VMS polygons (magenta) and roadside access (blue)

Approximate scale: ↔ 2 m

Figure 3.7.2: CIR image with VMS polygons (magenta) and roadside access (blue)

Approximate scale: ↔ 2 m

3.8 Manual labelling

The RGB images were jigsawed and written into an MS Excel spreadsheet to facilitate manual labelling, as shown in Figure 3.8.1. At this point I intended to try a vehicle detection algorithm, hence the labels chosen were 1: Vehicle present in image, 2: Parking possible, 3: No parking possible, 4: Not sure. We have 1329 images from the test area, and the spreadsheet format makes manual labelling very smooth. I was able to label about 250 images in under 30 minutes. I recruited some colleagues to help with the labelling and gave them about 250 images each. Hence each image was labelled twice, and the majority (~80%) were labelled by two different people.

Figure 3.8.1: Manual labelling spreadsheet example

Label	GRID	TOID	RGB_25cm_png_image
	TQ1980	1000001778697865	

Figure: 3.8.2 Confusion matrix comparing two rounds of manual labelling

label_tjf \ label_others	1	2	3	4
1	300	16	7	0
2	26	305	43	6
3	4	118	464	30
4	1	8	0	0

Comparing the two sets of labels, I found 80% agreement, see Figure 3.8.2. The largest disagreement, 118 images, or 9%, occurred between my 'No parking' label, *label_tjf*, (3) and my colleagues', 'Parking possible' label, *label_others*, (2). I was unable to give my colleagues any indication of size of the images, other than that they were all to the same scale, and they could use the images containing vehicles as a guide to whether a vehicle might fit in another image. I reviewed all disagreements and made a final decision, also eliminating the 'Not sure' label (4). Whilst reviewing it became clear that most of the 3-2 discrepancies occurred when the manmade surface area was borderline, although, I can only defend my decisions by proclaiming my greater familiarity with the data. It might seem strange that there is also disagreement (3%) in whether a vehicle is present in the image, however, with shadows and bins or small sheds in front gardens it can be surprisingly tricky to tell. The final reviewed labels showed 84% agreement with my colleagues' labels, and 94% agreement with mine (hardly surprising since I did the review). However, the increase in agreement with my colleagues' labels does demonstrate that I conceded in some cases, as well as overruled.

For my current method labels 1 and 2 both indicate parking, so disagreements between these labels are inconsequential. Taking this into account agreement increases to 84% between my colleagues' labels and my labels, and post-review increases to 87% and 96% between the reviewed labels, and my colleagues' and mine, respectively. Overall, this is sufficiently good agreement.

4. Experiments & results

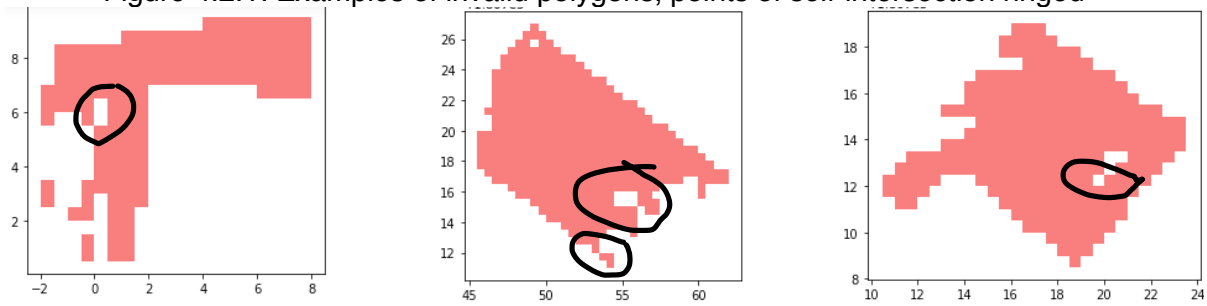
4.1 Data and tool familiarisation

I spent some time investigating the data and the python libraries to better understand what information I had available and how best to use the tools. For instance, I also had access to the OSMM Topographic Line layer, which contains only line segments delineating boundaries, as well as the OSMM Highways Network of roads. The python library OpenCV contains a huge range of image processing functions, including segmentation algorithms. However, these functions are written for RGB images, and employ techniques such as segmentation by colour gradient and averaging over neighbouring pixels. Since vegetation has special properties which are captured in the CIR images I decided to attempt segmentation by NDVI value first.

4.2 Maintaining tidy geometries

Spatial joins and intersections can have surprising effects on geometries. Intersections may include point intersections, colinear lines may be rendered as multi-linestrings, polygons may be rendered invalid (self-intersecting, see Figure 4.2.1), holes in polygons may be incorrectly orientated.

Figure 4.2.1: Examples of invalid polygons, points of self-intersection ringed



Axes scales in metres, 1 pixel ≡ 0.5 m × 0.5 m

Due to the frequency of these issues it was necessary to build checks into the pipeline. These included checking for validity, orientation, and geometry homogeneity, where expected. For example, creating the roadside linestring geometries per property unexpectedly produced point geometries and multi-linestrings that were actually colinear. I dealt with these by attempting to merge multi-linestrings, discarding any remaining point geometries. Multi-linestrings in general are a valid outcome, since a roadside edge may have a hedge interrupting the intersection with the manmade surface.

4.3 Filling holes

Many of the VMS polygons contained one or more holes. The histogram in Figure 4.3.1 shows the distribution of holes with areas under 10 m^2. The vast majority are under 1 m^2 in size. This corresponds to 4 pixels in the CIR images, and while this could be a solid plant-based obstruction it could also be passable. It seemed reasonable to allow some flexibility to account for overhanging foliage and areas of uncertainty, thus all holes with area less than or equal to 1 m^2 were filled.

Figure 4.3.1: Distribution of hole areas under 10 m²

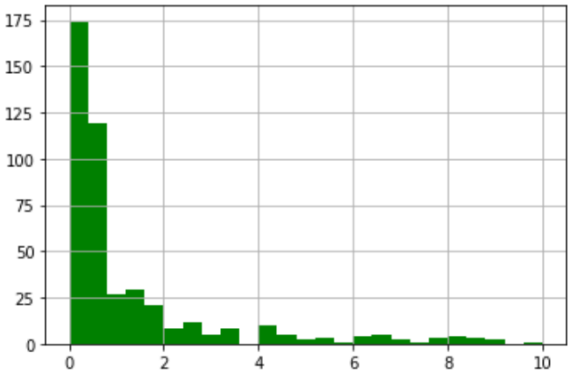
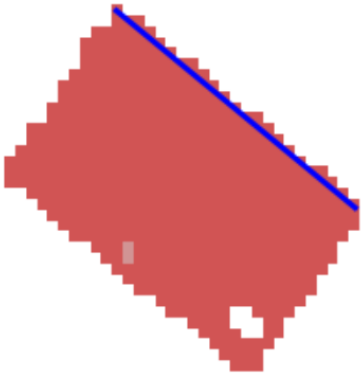


Figure 4.3.2: Example of a VMS polygon with 2 holes, with the smaller filled in



Scale: 1 pixel ≡ 0.5 m

4.4 Remove clear no parking areas

Before trying to fit parking sized rectangles it made sense to remove any of the 2626 VMS polygons that could clearly not accommodate a vehicle, by area and access width. The standard parking space in the UK is 2.4 m by 4.8 m, which has area 11.52 m². This is a generous size as it must accommodate everything from the Mini Cooper to the 'Chelsea tractor'. Since the VMS polygons are created from pixelated images it is reasonable to choose a cut off area that is an integer number of pixels. Rectangular shape, in ratio roughly 1:2, is also important and I reasoned that it is unlikely that borderline areas will be precisely the correct shape. Hence, I took the cut off area as 2.5 m by 5 m, or 12.5 m². The histogram in Figure 4.4.1 shows that over 1000 VMS polygons with area under 14 m² have area less than 2 m². Despite the large number of undersized VMS polygons, removing them only resulted in 37 TOIDs (properties) being classified as 'No parking possible'. This is due to the majority of properties containing other, larger, VMS areas. The undersized polygons tend to be very small and scattered, for examples see Figure 4.4.2.

Figure 4.4.1: Distribution of VMS areas under 14 m²

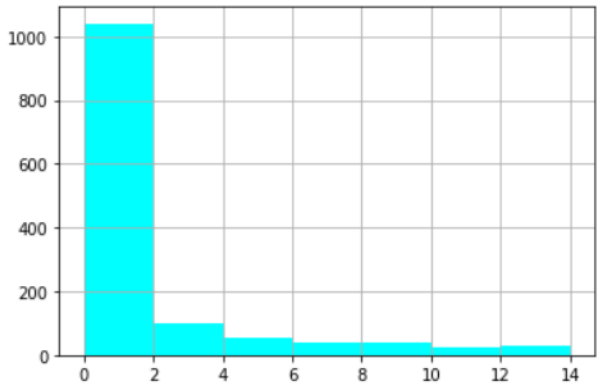
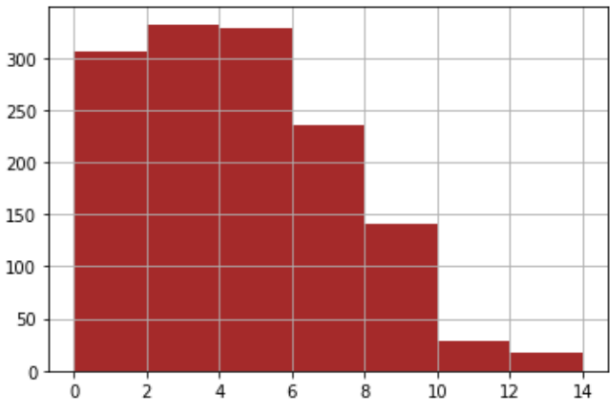


Figure 4.4.2: Undersized VMS polygons (red), corresponding garden polygon (green)

1 pixel ≡ 0.5 m × 0.5 m

Roadside access to the manmade surface is essential for parking. I classified any VMS polygons with roadside access linestrings less than the width of the parking space, 2.4 m, as 'no parking possible'. The histogram in Figure 4.4.3 shows the distribution of roadside access lengths, zero lengths have already been removed. Removing TOIDs where the roadside access is zero or too narrow classified 133 + 188 = 321 properties as 'no parking possible'.

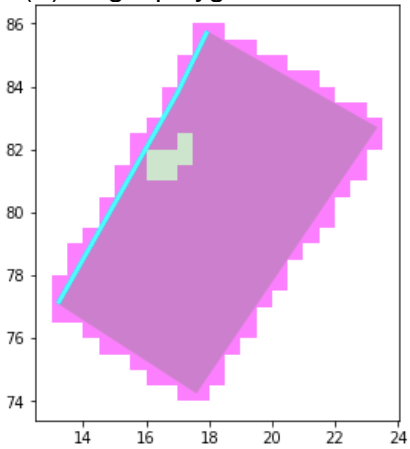
Figure 4.4.3: Distribution of (positive) roadside access linestrings lengths under 14 m



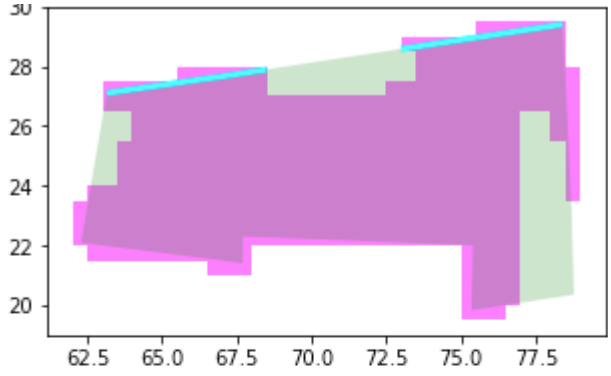
After classifying properties as 'no parking possible' if the VMS area is too small or the roadside access too narrow, we are left with 971 properties/TOIDs to check if a parking sized rectangle will fit. Figure 4.4.4 shows examples of the different combinations of number of VMS polygons and roadside access linestrings that are possible in properties that pass the area and roadside access filter.

Figure 4.4.4: Example properties that comprise sufficient VMS area (pink) and roadside access length (cyan). Combinations of single and multiple polygons and linestrings per TOID. Axes scales are in metres

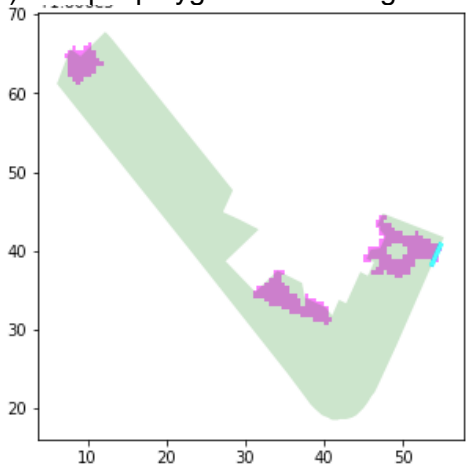
(A) single polygon and linestring



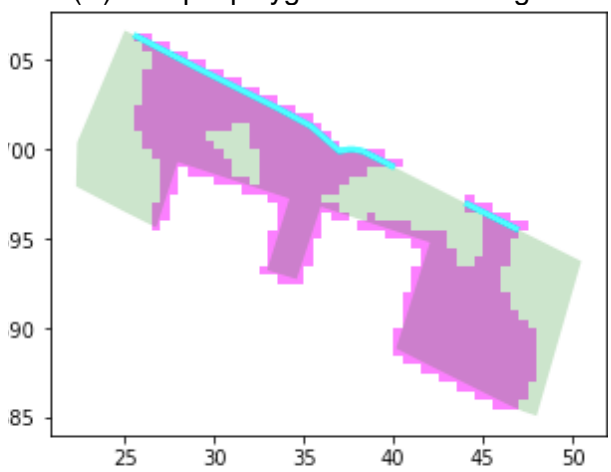
(B) single polygon and multiple linestrings



(C) multiple polygons and a single linestring



(D) multiple polygons and linestrings



4.5 Rectangle fitting

There many ways to try to fit a moveable rectangle of fixed dimensions into a fixed polygon. Determining the starting position and orientation of the rectangle are the main considerations. A brute force approach may try random start positions and multiple orientations; however, this quickly becomes computationally expensive. To limit the search, I focussed on fitting rectangles with a short edge aligned to the roadside access linestring. The roadside linestring as well as providing width of access also, in most cases, defines the orientation of the property with respect to the road. Furthermore, since a vehicle must cross the access line, it must fit through this area, even if it then parks further inside the property extent. Hence it is sufficient to check if a parking rectangle fits at the access edge. At this time I am only interested in identifying properties with the potential to park at least one vehicle, and am not attempting to count how many vehicles may be accommodated.

I wrote a function called *six_pts()* that takes a linestring as input and outputs six points interpolated from it. These points are arranged in pairs, 2.4 m apart, positioned at either end, and in the middle of the linestring. My next function *three_buff_lines()* takes these pairs of points, creates a linestring geometry from each pair and then buffers this linestring to 4.8 m. The shapely buffer function is a convenient way to extend a geometry to a certain distance around its perimeter. Buffering a straight line of length l to distance d, and specifying the cap style as 'flat', extrudes the line width d distance on both sides and perpendicular to it. It returns a rectangle of dimensions $l \times 2d$ and the original line bisects this rectangle. The intersection of these rectangles with the VMS polygons per TOID is then computed, the resulting polygons define the area of overlap. In the shapely framework it is possible to indicate each side of a line, however, I had no way to determine which side the property was on other than to compute a spatial intersection.

Figure 4.5.1 shows the same properties as in Figure 4.4.4, and additionally the three rectangle positions for each roadside linestring. Properties (C) and (D) are both corner properties, such that the rear garden is adjacent to the roadside. The rear garden VMS polygons in (C) do not intersect the roadside edge so no rectangle overlap occurs in these areas. In (D) the rear garden has a large manmade surface area, but the three positions of the rectangles have coincidentally aligned with significant shrubberies so that the overlap is not large enough to count as parking. A problem with aerial photography is demonstrated in (D) where a tree canopy obstructs the front garden view. A vehicle can clearly be seen in the image at coordinates (43, 90) but the overlap of the rectangles fitted is sufficiently decreased by the overhead foliage that this will likely be incorrectly classed as no parking possible.

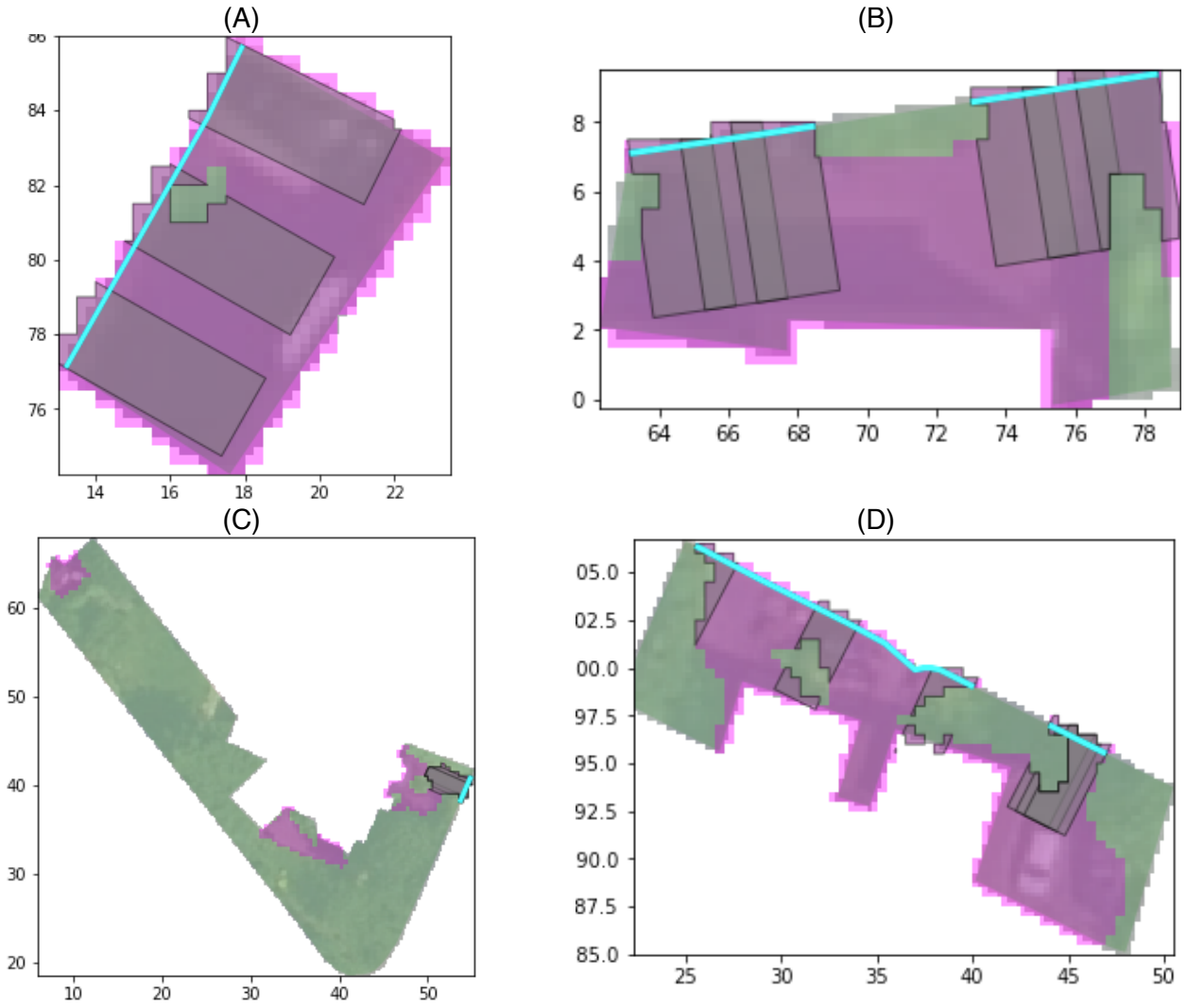
(A) and (B) demonstrate that the rectangles may, or may not overlap each other, depending on the length of the roadside linestring. I considered starting the rectangles at a small offset from each end of the linestring, to allow for likelihood of boundary vegetation. However, since I removed linestrings less than 2.4 m, my smallest linestring could be exactly 2.4 m, the width of a rectangle. Interpolating a point 2.4 m along a line that is too short would cause problems and make the algorithm more complex. So, for simplicity I kept the rectangles aligned with each end of the line.

The output from the rectangle fitting is a dataframe containing all the polygon intersections of each of the three rectangles, per linestring, per VMS polygon, per TOID (property). I will refer to these polygons as the rectangle overlap. This may include empty polygons, where no intersection occurs. The number of rows, r, in the dataframe output is given by

$$r = 3 \sum_{i=1}^{t} l_i \times p_i$$

where t is the number of TOIDs, l_i and p_i are the number of linestrings and number of VMS polygons for the ith TOID, respectively. I wrote a function based on this formula to check the algorithm was functioning as expected. For the 971 properties 3339 rectangles were fitted. This makes sense since the value should be at least three times the number of properties.

Figure 4.5.1: Example properties, VMS polygons (pink), roadside access (cyan), vegetation (green), three rectangles fitted for each roadside linestring, RGB raster image base
Axes scales are in metres

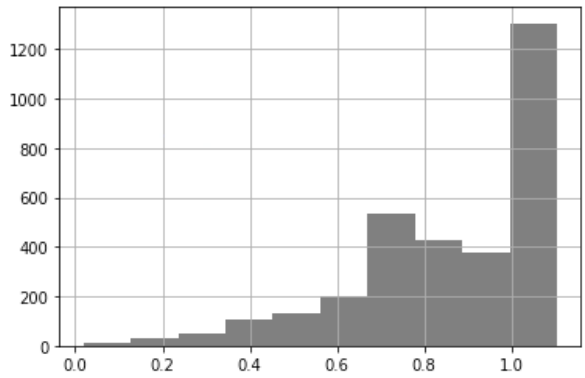


4.6 Validation

I validated the rectangle fitting algorithm against the manually labelled properties, having first mapped manual labels 1 and 2, which both indicate parking, to label 1. 'No parking' is mapped to 0. Note that this validation compares the labels of all 1329 properties. Of these, 971 properties are classified by the rectangle fitting algorithm and 358 properties are already classified as 'no parking' since they are undersized or the roadside access is too narrow. It is reasonable to include these together since the rectangle fitting algorithm as it is would not change any of these labels. However, if the roadside linestring were permitted to be less than 2.4 m and extrapolation allowed then it is likely that there would be a difference. I will return to this point later.

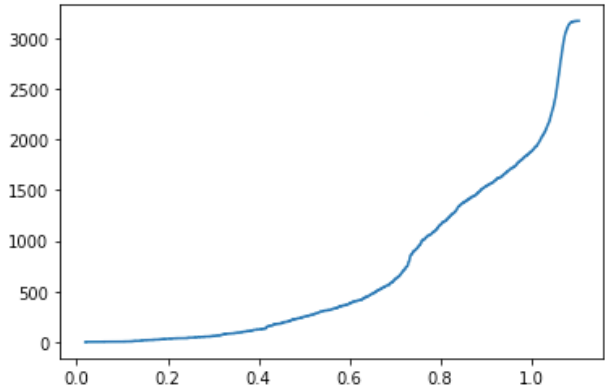
Originally, I had thought to label areas as 'parking possible' only if a rectangle fit completely. To allow for some flexibility I decided to determine a threshold based on the proportion of the rectangle area that overlapped the VMS area, above which a property would be classed as 'parking possible'. Having removed empty polygons from the data I calculated the proportion of rectangle overlap, which is the rectangle overlap area divided by the original rectangle area, $2.4 \times\ 4.8 = 11.52\ \mathrm{m}^2$, and plotted the distribution and cumulative frequency graphs. Removing the empty polygons left 3169 rectangles with positive area.

Figure 4.6.1 Distribution of area of rectangle overlap with VMS



Proportion of overlap

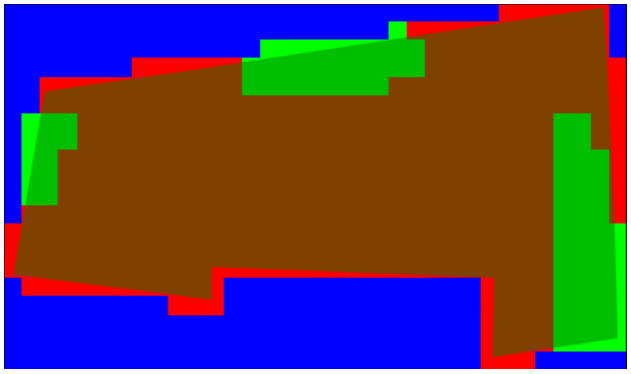
Figure 4.6.2 Cumulative frequency graph of area of rectangle overlap with VMS



Proportion of overlap

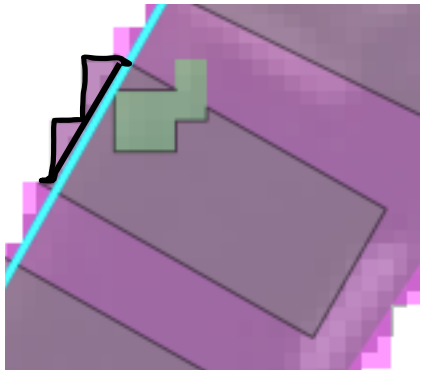
The final bar in Figure 4.6.1 shows that nearly half of the rectangles overlap by 100% or more. The occurrence of proportions greater than one is an artefact due to a combination of the jigsaw procedure and the line buffering step in the rectangle fitting algorithm. The jigsaw function includes pixels that 'touch' the boundary of the polygon, rather than including only those that are entirely contained by it. The roadside linestring buffering method creates a parking rectangle either side of the line. Hence, when these are intersected, a small area of the VMS polygons that lies outside the original cropped garden polygon is included. Figures 4.6.3 and 4.6.4 illustrate this.

Figure 4.6.3 Segmented jigsaw image, manmade surface (red), vegetation (green), non-garden (blue) and original cropped garden polygon overlaid



1 pixel $\equiv 0.5\ \mathrm{m} \times 0.5\ \mathrm{m}$

Figure 4.6.4 Rectangle overlap outside original cropped garden polygon outlined in black



Scale: rectangle short edge length is 2.4 m

The maximum rectangle overlap proportion in the data is 1.10, and the extra 10% corresponds to 1.15 m². The most extreme circumstance occurs when the property is perfectly aligned to the grid and pixel position, and only the edge of each pixel touches the intersection. This unlikely scenario provides an upper bound for the extra area of 4.8 pixels, or $2.4 \times 0.5 = 1.2\ \mathrm{m}^2$, which is less than 10% of a parking rectangle area, 11.52 m². This anomaly is acceptable as it is relatively small, spread over the width of a rectangle and effects all the data to some degree.

To decide an appropriate rectangle overlap proportion threshold I plotted the proportion of agreement, (true positives and true negatives), between the rectangle labels and the manual labels for all thresholds, as shown in Figure 4.6.5. This indicates an increase in agreement at the 80% threshold. Figure 4.6.6 shows the proportion agreement for thresholds of 80% and above. There is a shallow hump up to around 100% after which the agreement proportion drops away steeply.

Figure 4.6.5 Bar chart of proportion agreement for all rectangle overlap proportions

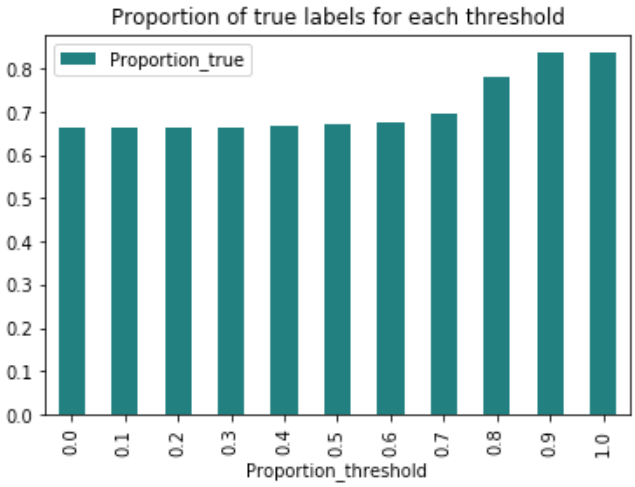
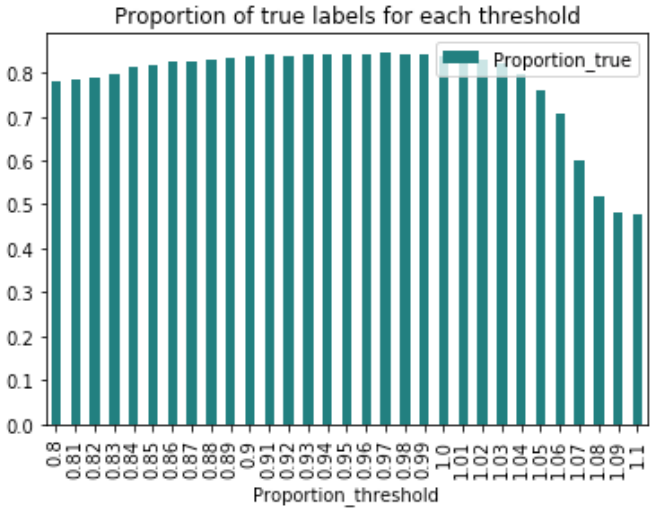


Figure 4.6.6 Bar chart of proportion agreement for rectangle overlap proportions of 80% and over



I found the maximum agreement occurred with the threshold for the rectangle overlap set to 97%. Figure 4.6.7 shows the gain or loss of correctly classified properties from one threshold to the next. Compared to the neighbouring values 97% is a distinct peak, we gain 6 correct labels over the previous threshold, and lose 8 at the next.

Figure 4.6.7 Change in number of properties (TOIDs) labelled correctly for thresholds with proportion true over 84%

	Proportion_threshold	Proportion_true	TOID_gain
0	0.93	0.84123	NaN
1	0.94	0.84123	0.0
2	0.95	0.84274	2.0
3	0.96	0.84274	0.0
4	0.97	0.84725	6.0
5	0.98	0.84123	-8.0
6	0.99	0.84048	-1.0

Overall there is very good agreement, above 84%, between the algorithm and the manual labelling for thresholds in the interval [0.93, 0.99]. If we take agreement at 80% this interval widens to [0.84, 1.03]. This is reassuring and implies the method is robust and not sensitive to small changes.

Figures 4.6.8 and 4.6.9 show confusion matrices comparing the rectangle algorithm labels to the manual labels for overlap threshold 0.97; Figure 4.6.9 is normalized. As noted above we have 84% agreement (true positives and true negatives). Disagreement is 16%, with 11% false negatives and 4% false positives. False negatives occur when the rectangle fitting algorithm labels the property as ‘no parking’, 0, and the manual label is ‘parking possible’, 1. The false negative count is nearly triple that of the false positives, hence the model underestimates the potential parking in the test area.

Figure 4.6.8: Confusion matrix of manual labels (manLab) and algorithm labels (recLab) at overlap threshold 0.97
0: no parking, 1: parking

recLab	0	1	All
manLab			
0	575	55	630
1	148	551	699
All	723	606	1329

Figure 4.6.9: Normalized confusion matrix of manual labels (manLab) and algorithm labels (recLab) at overlap threshold 0.97
0: no parking, 1: parking

recLab	0	1	All
manLab			
0	0.432656	0.041384	0.474041
1	0.111362	0.414597	0.525959
All	0.544018	0.455982	1.000000

As noted the analysis above compares labels of all 1329 properties, where 358 are classified as ‘no parking’ before the remaining 971 are checked by fitting rectangles. An analysis of only those properties where rectangles were fitted reveals that the relatively large number of false negatives, 148 (11%), can be attributed to the previous step. The peak threshold is unchanged at 0.97 since in each comparison the 358 pre-classified ‘no parking’ labels remain the same. Removing these properties improves the rectangle fitting agreement from 84% to 88%. The false negatives and false positives are now almost the same at 6.3% and 5.6%, respectively. See Figures 4.6.10 and 4.6.11.

Figure 4.6.10 Confusion matrix of manual labels and **rectangle only** labels at overlap threshold 0.97
0: no parking, 1: parking

recLab	0	1	All
manLab			
0	304	55	359
1	61	551	612
All	365	606	971

Figure 4.6.11 Normalized confusion matrix of manual labels and **rectangle only** labels at overlap threshold 0.97
0: no parking, 1: parking

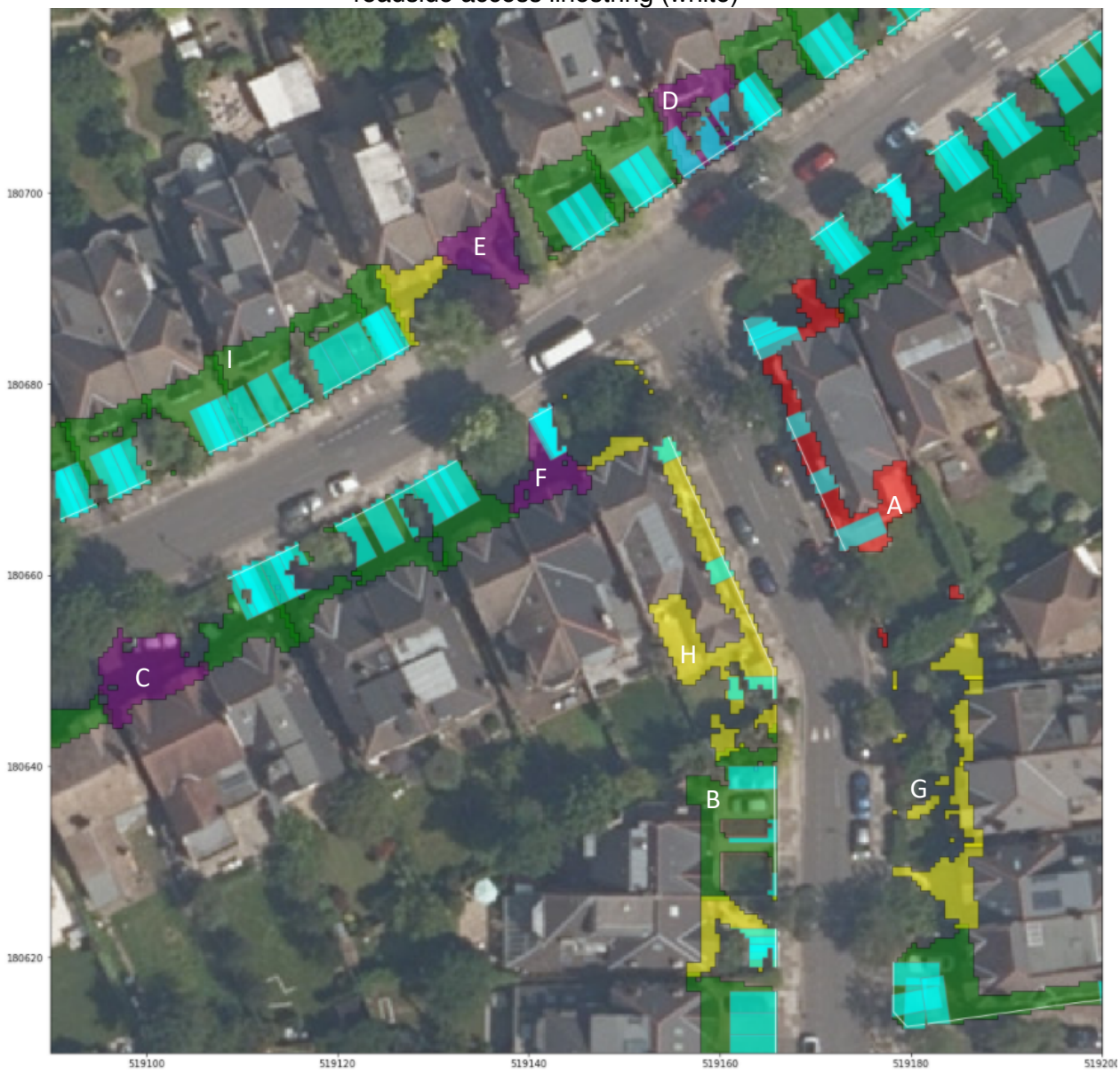
recLab	0	1	All
manLab			
0	0.313079	0.056643	0.369722
1	0.062822	0.567456	0.630278
All	0.375901	0.624099	1.000000

Figure 4.6.12 shows an area containing VMS polygons of all four combinations of true/false positive/negative, along with the parking rectangles, where fitted, and the roadside access linestring. Negative labels (yellow and purple) may not have rectangles fitted, as the pre-classification only identified ‘no parking’ properties. I have marked interesting features on the plot which I discuss below.

I suggest that a significant cause of false negatives is overhanging foliage, mostly tree canopy. This may reduce the appearance of both manmade surface area and roadside access. Properties C and E in Figure 4.6.12 demonstrate this. In each case the roadside access is completely obscured or significantly reduced by tree canopy.

Properties D and F are false negatives classified by the rectangle fitting algorithm. F indicates the algorithm’s failure to take into account the fact that a vehicle can turn corners, though the roadside access is also obscured by tree canopy. D illustrates how the roadside may not always be perpendicular to a property. The left most rectangle is very close to overlapping enough area to count as parking, but too much falls outside the boundary.

Figure 4.6.12 VMS areas: True Positive (green), True Negative (yellow), False Positive (red), False Negative (purple), parking rectangle overlap where fitted (cyan), roadside access linestring (white)



Axes in metres, distance from BNG datum

Property A is a corner property and the rectangle fitting has generated a false positive because the rear garden appears to have roadside access and contains a large patio.

Property B is a true positive, with a car parked perpendicular to the road visible in the image. This illustrates the main assumption of the rectangle fitting algorithm, that the vehicle will either park perpendicular to the road or transit that space en route to parking further within the property bounds.

Properties G and H are both true negatives, the former is classified due to small area and lack of roadside access and the latter by the rectangle fitting algorithm. H is also a corner property and shows how these may have very long roadside linestrings, meaning that the three rectangles fitted occur at wide intervals. For these situations the outcome is somewhat hit and miss. H appears to have some rear garden manmade surface that, had the rectangle aligned with it, could have generated a false positive.

Finally, property I is an example of an artefact of the aerial photography that is present throughout the data where the property is not photographed from directly above. This makes the roof of the building overhang the ground level footprint of the front garden polygon. Since the roof is a manmade surface it may overrepresent the VMS area. Overhanging roof could only cause an incorrect labelling (a false positive) if the obscured area is necessary for a parking rectangle to fit. That is, if the roof overhang obscures vegetation, and there is visible manmade surface next to the roadside, so that it appears as one contiguous manmade surface. However, this seems an unlikely arrangement. G shows that in a 'no parking' front garden the roof level manmade surface is disconnected from the roadside and so is discounted. Furthermore, if the roof level manmade surface simply obscures ground level manmade surface, then it makes no difference.

4.7 Results

Out of the test set of 1329 residential dwellings in a 1 km square suburban area, 606 (46%) were classified as 'parking possible' and 723 (54%) as 'no parking' by the algorithm I developed. This showed very good agreement (84%) with the manual labelling results which classified 699 (53%) as 'parking possible' and 630 (47%) as 'no parking'. The model slightly underestimates the number of 'parking possible' properties.

5. Discussion

The results of this project are very encouraging, and this method is shown to be at least as good as manual labelling. The algorithm labelled test set of 1329 properties showed very good agreement (84%) with the manual labelling. The main findings include possible refinements and improvements that could be implemented. These are outlined below.

The main weak point of the process is that it is susceptible to tree canopy which may obscure the manmade surface and/or the roadside access. This is also a problem for manual labelling, although if only the roadside access is obscured then a human may discern parking further within the property bounds. This recommends ameliorating the roadside access issue first by reducing the roadside linestring cut-off value and allowing extrapolation of the line when creating the first two vertices of the rectangles to fit. Currently the cut-off value is 2.4 m and interpolation only is allowed. This change would increase the number of properties that are fitted with rectangles and allow for those rectangles to have vertices outside the manmade surface area. Alternatively, I could simply reduce the size of the parking sized rectangle, perhaps make it narrower. A more involved approach may be to attempt to identify trees as a roughly circular area of uniform colour and estimate the trunk location. If the trunk is on the property it may still present an obstacle. It is not recommended to grow trees over 10 m high close to a building so in many cases large trees are grown on the pavement. Hence, they are more likely to obscure the roadside access than all of the manmade surface in the garden.

In my research I investigated how to check if a fixed polygon contains another moveable polygon. I did not come across anything particularly useful. Once a position is decided for the moveable polygon, in this case rectangle, it is trivial to determine if it fits. However, how does one decide these positions? Considering that a random approach is too costly. I chose to align the rectangle with a particular edge of the polygon. However, I think it would be useful to explore other ways of establishing starting positions, as sometimes the roadside access is partially obscured or bounded by vegetation that is actually passable. Now that a good range of proportion overlap thresholds have been established, I can stop the rectangle fitting as soon as a satisfactory position is found. Hence, I can generate more positions to check. A grid search algorithm might work where the starting vertex of the first n rectangles is on the roadside line at some interval and then moves one pixel inside the polygon and continues the search at same interval and parallel to the roadside line. Accounting for turning is more complex: At each starting position I could try different angles to the roadside linestring. Additionally, there may be merit in developing a search algorithm that learns from the previous fit by calculating the proportion overlap, the location of the centroid and choosing the next starting position accordingly.

Finally, improvements could be made to the segmentation method. It would be useful to explore converting the RGB image to HSV colour palette, as demonstrated in the GreenSpaces project (Bonham, 2019). Then general image processing techniques could be used to improve the image before calculating the NDVI, or other vegetation index value.

6. Conclusion & future work

The scope of this project is bold and although I have not yet produced UK wide parking potential estimates I am very pleased with the results as a viability study. So far I have spent three months on this project, whilst also working on another project of similar scale. For comparison the ONS GreenSpaces project, which did scale to the UK, involved three people over six months, approximately six times my resource. As a learning opportunity this project has been extremely worthwhile and intellectually stimulating, and I hope to continue with it to its full conclusion.

6.1 Implications and impact

The UK government is committed to phasing out the manufacture of petrol and diesel cars and vans such that by 2040 all new cars and vans will be effectively zero emission (DfT, 2018). At the end of 2019 there were 36 million licensed cars and LGVs vehicles in the UK, of which 269 thousand (0.7%) were Ultra Low Emission Vehicles (ULEV)(DfT, 2020). Thus, a great many more electric vehicles are expected to enter the UK stock over the next 20 years. This will have a huge impact, not just on emissions, but also on vehicle owners and the UK electricity network. As noted above it is expected that most people will charge their EV at home overnight. To support EV chargepoints of different power requirements it is essential to anticipate where they will be situated to ensure that the network is robust. It is expected that public chargepoints will be more powerful (and therefore charge more quickly) than private chargepoints. Hence, the proportion of private versus public chargepoints required in an area will inform investment in substations and cabling.

The result that the proportion of residential dwellings with off-street parking is about 46%, within a 1 km square suburban area, may be used cautiously to provide very rough estimates for similar areas. (Suburban areas are identifiable in various government data sources). Overall, areas with a high proportion of available off-street private parking will require private investment (possibly including government incentives) in low power overnight chargepoints. Whereas areas with a low proportion of private parking will require a mixture of low and high powered public chargepoints, and potentially improvements to the local substations.

This project successfully demonstrates the viability of this method for identifying off-street parking on residential sites. Extended to the UK these data will constitute the first comprehensive dataset on this subject and contribute vital information to UK EV infrastructure planning.

6.2 Future work

The next step, beyond refinements suggested above, is to scale this method to identify off-street parking for every residential dwelling in the UK. The Lower Super Output Area (LSOA) is a geospatial statistical unit that contains a minimum population of 1000 people, with a mean of 1500. Therefore, it should have similar numbers of residential dwellings as in my test area. I used the 1 km square grid TQ1980 for my test area, however it is more useful to classify parking potential within recognised statistical areas. It is also more practical, since an LSOA is small enough to not overlap too many of the 1 km square grids in the aerial photography data, and since each property only falls in one LSOA we do not have to account for properties straddling the boundary of the 1 km square grid. I will develop a method of collecting the AB+, TA, LRII and APGB data for each LSOA, and perform the algorithm per LSOA. Having investigated the range of proportion of rectangle overlap that produces the best agreement with the manual labelling it is now possible to set a threshold and stop fitting rectangles as soon as one is fit that meets the threshold. This would save computing time and memory.

An alternative scaling approach is to use the current labelled data to train a machine-learning algorithm. I would need to include more urban and more rural areas, and perhaps do more manual labelling. The rectangle fitting algorithm has performed well in suburbia, although it is susceptible to tree canopy. It is not designed to account for underground or other covered parking as might occur more urban areas. Nor has it been tested in a rural area where parking may likely occur on non-manmade surfaces such as grass. However, it would be useful and interesting to attempt a combination of manual labelling and rectangle fitting labelling to prepare training sets for machine-learning classification.

This project has investigated parking potential using remote sensing techniques to establish the existing (within the last few years) manmade surface area within a roadside garden of a property. A simpler approach is to use only the AB+ and TA layer data, and to use the rectangle fitting algorithm on the full cropped garden polygons, regardless of whether there is vegetation present or not. This would establish a maximum number of properties with potential off-street parking, since a garden may be converted to parking area. Scaling this would also be easier as the aerial photography, and hence image processing, is removed.

References

APGB. (2020a). APGB Aerial Photography User Guide. Retrieved July 25, 2020, from APGB Support Knowledge Base website: https://support.apgb.co.uk/en/article/apgb-aerial-photography-user-guide

APGB. (2020b). APGB Colour Infrared (CIR) User Guide. Retrieved July 25, 2020, from APGB Support Knowledge Base website: https://support.apgb.co.uk/en/article/apgb-colour-infrared-cir-user-guide

Bonham, C. (2019). Green spaces in residential gardens. Retrieved July 24, 2020, from Data Science Campus Blog website: https://datasciencecampus.ons.gov.uk/projects/green-spaces-in-residential-gardens/

DfT. (2018). *The Road to Zero*. Retrieved from www.gov.uk/dft

DfT. (2020). *Vehicle Licensing Statistics: Annual 2019*.

Emu Analytics. (2018). *ON-STREET EV CHARGER REQUIREMENTS PRODUCT SHEET*.

HM Land Registry. (2020). HM Land Registry INSPIRE View Service Metadata. Retrieved August 18, 2020, from https://data.gov.uk/dataset/82f01b50-51e8-484c-82c3-40207557c9d5/hm-land-registry-inspire-view-service-metadata

Jacob Rus. (2010). HSL and HSV. Retrieved August 17, 2020, from Wikipedia website: http://upload.wikimedia.org/wikipedia/commons/a/a0/Hsl-hsv_models.svg

OLEV. (2011). *Making the Connection The Plug-In Vehicle Infrastructure Strategy*. Retrieved from www.dft.gov.uk

Ordnance Survey. (2016a). *OS MasterMap Topography Layer Technical Specification*. 61–62. Retrieved from https://www.ordnancesurvey.co.uk/docs/user-guides/os-mastermap-topography-layer-user-guide.pdf

Ordnance Survey. (2016b). *Using the National Grid*. 1–2.

Ordnance Survey. (2018). *AddressBase Plus Technical Specification*. *117*(803), 366–366. https://doi.org/10.1525/curh.2018.117.803.366

Xue, J., & Su, B. (2017). Significant remote sensing vegetation indices: A review of developments and applications. *Journal of Sensors*, *2017*. https://doi.org/10.1155/2017/1353691

Zhou, S., & Simmons, J. (2019). *From Minkowski Sum to Concave Hull : Two Case Studies of Open Source Development at Ordnance Survey*. (April).

Appendices

What follows is a list of the scripts containing all the code for this project available at this link: https://drive.google.com/drive/folders/1QUvy-ubrd_YgELhwKwrkM0QIuIV_VZZ2?usp=sharing

Jupyter notebook filename	Description
ppp01_os_openmap_local_chargepoints.ipnyb	Explores OS OpenMap Local and registered (ie not all) public EVCP locations
ppp02_osmm_gpkg.ipnyb	Explores the different layers and fields of the OSMM Topographic Layer geopackage file
ppp03_ap_tjf.ipnyb	Explores the channels of the APGB RGB and CIR tagged image format (tif) files
ppp04_EalingTopoTile1980.ipnyb	Explores BNG TQ1980 1 km grid, in Ealing Borough, subset of OSMM Topographic Layer. Area and Line layers only, clipped in QGIS.
ppp05_addressBasePlus.ipnyb	Explore AB+ data
ppp06_linkAllShapesWithinPropertyExtent.ipnyb	Establish link between all shapes within a LRII property extent
ppp07_jigsawAerialPhoto.ipnyb	Use jigsaw function to cut out individual RGB images
ppp08_jigsawNearInfraRedImages.ipnyb	Use jigsaw function to cut out individual CIR images
ppp09_skimageExperiements.ipnyb	Explore Scikit Image library
ppp10_LineStringsAndCroppedGardens.ipnyb	Create the cropped garden polygons and corresponding roadside linestrings
ppp11_jigsawCroppedGardens.ipnyb	Use jigsaw function with cropped garden polygons on RGB and CIR images
ppp12_NDVI.ipnyb	Explore NDVI value and plot
ppp13_segmentAndVectorize.ipnyb	Use NDVI to segment CIR jigsawed images and vectorise result, fix invalid polygons
ppp14_manualClassification.ipnyb	Create MS Excel spreadsheet for manual labelling
ppp15_fitThreeRectangles.ipnyb	Run algorithm to fit three rectangles per roadside linestring per VMS polygon per TOID
ppp16_validationThreeRectangles.ipnyb	Validate rectangle algorithm against manual labelling