Cultural Heritage

Aligning archive maps and extracting footprints for analysis of historic urban environments

S.D. Laycock *, P.G. Brown, R.G. Laycock, A.M. Day

University of East Anglia, Norwich NR4 7TJ, United Kingdom

ABSTRACT

Archive cartography and archaeologist's sketches are invaluable resources when analysing a historic town or city. A virtual reconstruction of a city provides the user with the ability to navigate and explore an environment which no longer exists to obtain better insight into its design and purpose. However, the process of reconstructing the city from maps depicting features such as building footprints and roads can be labour intensive. In this paper we present techniques to aid in the semi-automatic extraction of building footprints from digital images of archive maps and sketches. Archive maps often exhibit problems in the form of inaccuracies and inconsistencies in scale which can lead to incorrect reconstructions. By aligning archive maps to accurate modern vector data one may reduce these problems. Furthermore, the efficiency of the footprint extraction methods may be improved by aligning either modern vector data or previously extracted footprints, since common elements can be identified between maps of differing time periods and only the difference between the two needs to be extracted. An evaluation of two alignment approaches is presented: using a linear affine transformation and a set of piecewise linear affine transformations.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

Archives storing large collections of maps and sketches present invaluable information concerning the layout and changes that have occurred to many historic towns and cities throughout the world. Ordnance Survey maps in the UK have been produced since the 18th Century, detailing building locations. Sanborn maps in the USA have been produced for American cities since the mid-19th Century and contain information far beyond the layouts of the buildings, as they were typically used for fire insurance purposes. However, there are limited means to compare the building layouts in order to understand the changes that have occurred to an environment over time. Often labour intensive digitisation of the maps is required to obtain vector data representing the building footprints through the use of software tools such as ArcGIS [1]. In this paper, techniques are presented which can semi-automatically extract building footprints from digital images of archive maps, resulting in vector data representing the footprints. Once the footprints have been extracted the environment may be reconstructed using software products such as CityEngine [2]. Fig. 1 illustrates a set of building footprints, shown top left, which have been imported into CityEngine to create a three-dimensional building environment based on shape grammar rules for Pompeii, shown at the bottom. Alternatively, digital artists may create a collection of three-dimensional models which may be used to populate the scene [3].

Historic maps often exhibit inaccuracies in scale and consequently incorrect reconstructions may result, if they are used directly. One approach to reduce these problems is to align the image of the historic map to modern data of an environment, ignoring changes to the contours of the landscape. For historic locations this is achievable through matching corresponding features on an historic map to modern vector data. Temples, churches and castles which still remain are good features. In some cases a linear affine transform of a map may result in a close match, however, in other cases a piecewise linear affine transformation provides a better alignment. Products such as MapAnalyst [4], may also be used to achieve non-linear transformation, although undesirable warping may result. Once a sufficient alignment has been determined the footprints are extracted. The techniques presented in this paper enable the user to align sources using both a single linear mapping and piecewise linear affine transformations.

By extracting the footprints from maps of an area from several time periods, one may perform further analysis on the changes to an environment. The techniques developed are able to compare the result of a previous extraction, or current vector data, with a map from a different time period to determine the areas that have

---

* Corresponding author. Tel.: +44 (0) 1603 593795.
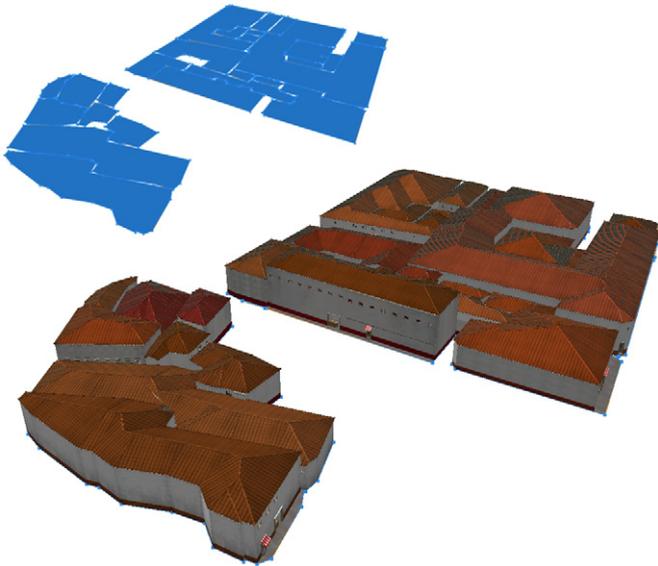 *E-mail address:* sdl@cmp.uea.ac.uk (S.D. Laycock).

Fig. 1. Building footprints, shown top left, have been imported into CityEngine to create the virtual environment shown at the bottom.

changed. Only the areas exhibiting change need be worked on further to obtain the complete extraction of the building footprints at different time periods. The next section covers relevant previous work.

2. Previous work

Constructing three-dimensional models of existing cultural heritage sites has received significant attention in two areas, namely laser scanning [5–7] and photogrammetry [8]. Whilst these approaches have been used extensively to record, measure and preserve cultural heritage sites they are only capable of displaying the current state of the environment. This paper considers the reconstruction of sites that no longer exist in their entirety, which poses a significant challenge in terms of obtaining an accurate representation of the area.

Procedural modelling techniques have been developed to create large environments based on building layouts. Muller et al. [2] reconstructed the city of Pompeii using a structured grammar. However, only a small section was reliably aligned to the manually defined building footprints. The remaining buildings were procedurally generated amongst the street network based on techniques presented in [9]. An approach is required to automatically extract the building footprints to facilitate the technique's application to large scale environments. Recently, CityEngine was applied to increase the fidelity of the laser scanned plaster of Paris model for Rome Reborn [10]. However, in this work, the building footprints were created by converting the initial scanned 3D model into a coarse representation, which could be input into the CityEngine system.

A variety of methods have been developed for extracting building footprints from high resolution aerial images [11–13]. However, typically these methods are not directly applicable to historic maps which are often of low quality. Suzuki and Chikatsu [14] investigated the automatic extraction of building plots from archived maps depicting Kawagoe, Japan in the Edo period. Their approach utilised the map to obtain the position of rectangular footprints by automatically extracting the four corners. This approach is limited in that it only recovered rectangular footprints. Removing this limitation, Laycock et al. [15] considered recovering building footprints represented by non-intersecting

closed polygons. However, this introduces further complications, since a polygon extracted from an image may have unit length edges. These require refinement such as the removal of near collinear points or the complete replacement with the largest oriented empty rectangle that can fit within the polygon.

Shimizu and Fuse [16] investigated techniques to compare changes to land-use in the Edo period. They presented a piecewise linear affine transformation strategy to align maps from different time periods. This technique is employed in this paper, however, we go further to use the different maps to aid in the extraction of building footprints to facilitate the analysis of changes to an urban environment. The next section describes the approach to extract the building footprints from an image of a map.

3. Extracting footprints from archive maps

In order to rapidly extract many building footprints from archive maps, representing an urban environment, an automatic or semi-automatic process is required to avoid the often prohibitively time consuming manual alternative. This section describes the process which aims to identify and extract the set of building footprints from an image of an archive map. The first step towards footprint extraction is to determine a location interior to each individual building on the map. Later stages will extract a footprint for each interior location. The representation of the buildings varies dependent on the style of the map in terms of colour, shading and building plot groups. If the buildings on the map are shaded with a solid colour, used uniquely for the buildings, then locations can be determined automatically. This can simply be achieved by searching for pixels of a specified target colour; once one is found then surrounding pixels in each direction can be tested. If these surrounding pixels, within a reasonable threshold, are also of the target colour then this location is likely to reside inside a building so it can be used as the start point for an automatic extraction process. In maps where buildings are not coloured uniquely, a monochrome map for example, a user may be required to click one point inside each building shape in order to provide a start point for the automatic extraction process.

Once the start point has been found it may be used as an initial seed point for a modified floodfill algorithm which 'fills' a point only if all pixels in a surrounding window are also of the target building colour. This constraint restricts the floodfill and prevents it from escaping through small gaps in building walls, but also means it does not fill precisely to the edges of the given area. If the floodfill area is larger than a given threshold it will be discarded and no further extraction will take place on the area.

A boundary walk is performed on the result of this floodfill to obtain a vectorised shape. Due to the fact the floodfill was constrained, the vectorised shape will not correctly represent the size of the footprint in the image. Therefore, this vectorised shape is used as the input to an active contour model which will then be expanded until it hits the black pixels that define the footprint edge. Adjusting thresholds of the active contour model allows control over the way it expands (e.g. by adjusting elasticity, it is possible to control the tendency to escape through small gaps or fit around sharp corners) and therefore small errors in the image would not cause problems as they would for a standard floodfill algorithm.

The results of fitting the active contour model are subsequently put through a line smoothing algorithm to simplify the shape and remove potential aliasing effects. Nearby points under a small threshold are merged together to further simplify the vector shape. Smoothing is handled by a two-stage process:

a McMaster [17] smoothing algorithm is used, followed by the removal of any collinear points that remain.

To improve the extraction further, enhancements to the algorithm can be made by generating an image mask by detecting edges in the original map image. Each point in the footprint can then be tested against the image mask to find out if the point lies along an edge on the map. Points that are not on an edge can be corrected by moving the point to a nearby corner or, failing that, the closest edge. The 'Fast' library [18] is used on the image mask to detect corners. Fig. 2(a) illustrates a portion of an image containing building footprints and Fig. 2(b) shows the image mask generated from an edge detection algorithm. The corners have been automatically identified with the 'Fast' library and are illustrated with black points.

Points are only moved if the move to the new position is deemed to be valid (see Fig. 3). A move is deemed valid if the new position does not cause either of the lines connected to the point to cross an edge on the image mask. If the connected lines were to cross edges then it is likely that the new point location represents a nearby corner of a surrounding building and the move would be judged invalid. The complete process for considering when to move a point is outlined in Fig. 4.

```
0 Begin Procedure: movePoint
1     IF edge pixels not found under point THEN
2         findListNearbyCorners(thisPoint)
3         FOR each nearby corner, starting from nearest
4             IF distanceToCorner < threshold THEN
5                 IF movingPointIsValid() THEN
6                     Move point to corner
7                     RETURN
8                 ENDIF
9             ENDIF
10        ENDFOR
11        IF point not yet moved THEN
12            findNearestEdge(thisPoint)
13            IF distanceToNearestEdge < threshold THEN
14                IF movingPointIsValid() THEN
15                    Move point to nearest edge
16                ENDIF
17            ENDIF
18        ENDIF
19    ENDIF
20 End Procedure: movePoint
```

Fig. 4. Procedure for attempting to correct points of extracted buildings that do not lie on a building edge. The test for 'movingPointIsValid()' carries out the process illustrated in Fig. 3.

When extracting angular footprints some sections may not be accurately recorded, due to poor image quality. To identify when these problems occur each line of the footprint is tested to check if it follows an edge or if it is crossing through the interior of a building. If it does not follow an edge then a recursive process is run on the line: it is split in two parts, and the newly created centre point is moved to the nearest point on an edge in the image. The test is repeated on the two newly created lines. The recursion continues until no change is required, or until the line length is below a minimum threshold. This process may have created collinear points, so a final removal of these is performed to straighten lines and to help to simplify the footprint shapes. When all processing is complete, footprints are checked for self-intersections to enable invalid building footprints to be discarded.

After a complete set of footprints has been extracted, a final optional processing step may occur. A building will often neighbour several others, but footprints extracted from a map are unlikely to join up precisely (they may for example be separated by the width of the black line in the image that defines their shape). If the user would prefer not to leave gaps between neighbouring buildings, an attempt can be made to merge neighbouring footprints. For each point X in a building, a list of nearby points is generated. All points in the scene are divided into a grid in order to speed up this process by minimising the number of tests to be done. The point with the smallest distance to X is the merge candidate, Y. If the distance between point X and Y is below a certain threshold, they will be merged. This is achieved by moving both points to a midpoint of the line between these two points. Fig. 5 shows an example of the merge process being carried out on two buildings.

The merging process is currently limited to considering two points joining together. Therefore, in cases such as a cross (+), results may be sub-optimal. This, and other complex scenarios, should be considered for future work.

The next section describes how vector data may be used to evaluate and to improve the efficiency of the footprint extraction methods presented in this section.

4. Using vector data to aid in the extraction of building footprints

Extracting the footprints from an archive map results in vectorised shapes which may be utilised as the basis of generating
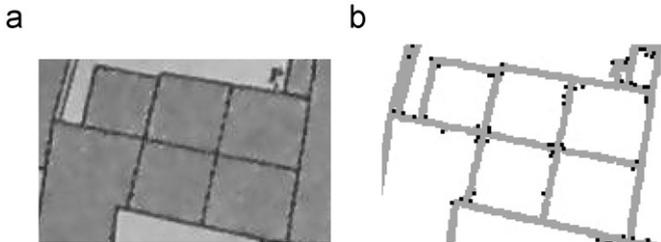

Fig. 2. A portion of an image of a map showing building footprints in black (a). (b) illustrates the same portion of the image with its edges detected and shown in grey. Corners are automatically detected and are marked here with black points.
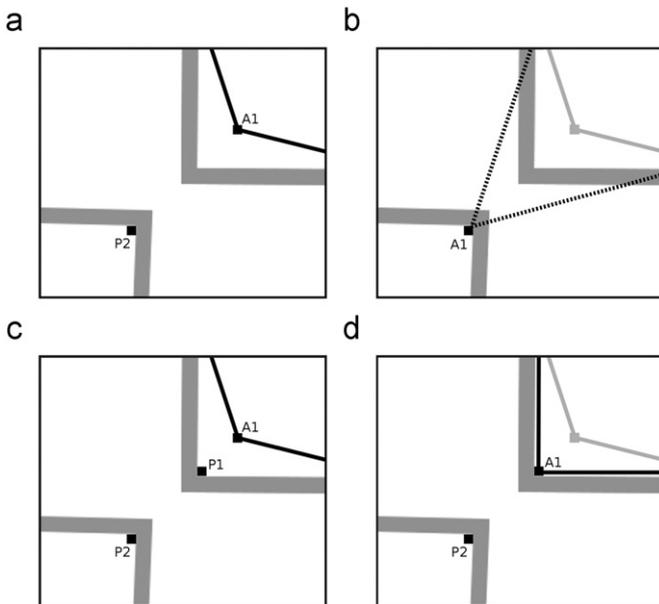

Fig. 3. A series of diagrams to illustrate attempts at correcting points and determining whether this move is valid. Image (a) shows an example point, A1, a nearby point (P2) that has been detected as a corner point in the image, and edges in the image, shown in grey. Adjusting point A1 to point P2 would cause the connected lines, which represent building walls, to cross one or more edges (b). This would not be a valid move. Consider if a nearer corner point, P1, had also been detected in the image(c). Moving point A1 to point P1 results in a valid move, because doing so does not cause either connected line to cross any edges in the image (d).
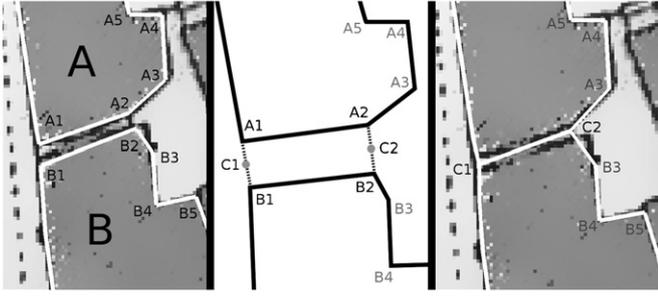
Fig. 5. The left image shows the points to be considered for the merge process from two example buildings, A and B. The middle image shows points that were identified for merging (A1–B1 and A2–B2) and how they will be merged. When processing point A1 a search for the nearest point, B1, was undertaken. The distance between A1 and B1 is less than a given threshold, so they will be merged. Point C1 is the midpoint of a line between A1 and B1. The merge process will move both points A1 and B1 to point C1, hence merging the two buildings and closing the gap between the two buildings. The same is true of the set A2, B2 and C2. The right image shows the results of the merging process.

a three-dimensional reconstruction of the environment. However, for an historic reconstruction the accuracy of the result is significant. This section discusses how accurate modern vector data may be used to assist in evaluating the results of the footprint extraction. To achieve this, one must be able to align the modern data set to the archive map. The next subsection discusses how this alignment may be accomplished.

With the ability to align data sets comes the possibility to reduce the footprint extraction times for archive maps of the same area. For example, assume a map depicting an area in 1850 has already been processed and we are given a second map of the same area depicting 1900. One may align the two maps and then undertake an analysis such that only the differences between the two need to be extracted. Section 4.2 details how this can be achieved.

4.1. Aligning data sets

To align one map to another we must find the transformation matrix which is capable of transforming coordinates on one map into the coordinate system of the other. In this work the transformation required to map vector based data, Input 1, to an image depicting an archive map, Input 2, is determined. Once the footprint extraction has been done on Input 2 the vectorised footprints may be transformed back to the coordinate system of Input 1 for further analysis or evaluation. The transformation is calculated in this way since it is faster to compute, as it avoids the complexity of transforming all image pixels where additional filtering would be required.

The simplest alignment process involves determining one transformation matrix for the entire map which considers non-uniform scale, rotation and translation of the coordinate system. This process is initiated by the selection of several pairs of corresponding points, typically using landmark buildings such as churches and temples. These locations currently must be manually defined by the user clicking on sets of corresponding points. Consider one pair of corresponding points where (x_1, y_1) is a point on Input 1 and (x'_1, y'_1) is the corresponding point on Input 2. We can define two equations to express the corresponding point in terms of the coordinate (x_1, y_1), as follows:

$$x'_1 = m_{11}x_1 + m_{12}y_1 + m_{13}$$
$$y'_1 = m_{21}x_1 + m_{22}y_1 + m_{23} \quad (1)$$

The unknowns in Eq. (1), m_{11}–m_{23} represent the transformation comprising rotation, translation and non-uniform scale. To determine the transformation required for the alignment at least

three pairs of corresponding points must be defined and these may be written in matrix form, as shown in Eq. (2):

$$\mathbf{Ax} = \mathbf{b} \quad \begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{21} \\ m_{22} \\ m_{23} \end{pmatrix} = \begin{pmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \end{pmatrix} \quad (2)$$

$\mathbf{A}$ is a $m \times n$ matrix, where n is the number of unknowns (in this case six) and m is equal to twice the number of the corresponding point pairs. In this case $m=n$, since three pairs of corresponding points are used. We can perform a factorisation of $\mathbf{A}$ using singular value decomposition to result in Eq. (3), where $\mathbf{W}$ is a diagonal matrix which can be stored as a vector of n values, $\mathbf{U}$ and $\mathbf{V}$ are both square matrices of dimension n and have orthogonal columns ensuring that each matrix multiplied by its transpose results in the identity matrix.

$$\mathbf{Ax} = \mathbf{b}$$

$$\mathbf{UWV}^T\mathbf{x} = \mathbf{b} \quad (3)$$

Using some simple linear algebra we may solve this equation to obtain the six unknown values in $\mathbf{x}$, as shown in Eq. (4):

$$\mathbf{UWV}^T\mathbf{x} = \mathbf{b}$$

$$\mathbf{WV}^T\mathbf{x} = \mathbf{U}^T\mathbf{b}$$

$$\mathbf{V}^T\mathbf{x} = \mathbf{W}^{-1}\mathbf{U}^T\mathbf{b}$$

$$\mathbf{x} = \mathbf{VW}^{-1}\mathbf{U}^T\mathbf{b} \quad (4)$$

Equation (5) shows how the six values may be used to form the transformation matrix:

$$\mathbf{T} = \begin{pmatrix} m_{11} & 0 & m_{12} & m_{13} \\ 0 & 1 & 0 & 0 \\ m_{21} & 0 & m_{22} & m_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5)$$

By clicking additional pairs of corresponding points on the inputs it is possible to verify the accuracy of the transformation matrix used for alignment. Each point from Input 1 may be transformed to the coordinate system of Input 2 and the Euclidean distance between the transformed point and the original point clicked on Input 2 may be determined. The average Euclidean distance details the overall error in the alignment.

Some historic maps suffer from inconsistencies in scale or distortions and therefore a single mapping for the entire map will not lead to adequate results. To prevent significant distortion that can occur the inputs may be divided into a group of corresponding triangles, where each triangle will be transformed by a unique transformation matrix. This process starts, as before, with a set of user-defined corresponding point pairs evenly distributed over the inputs. A Delaunay Triangulation [19] is then calculated using the points on one input and the same connectivity of the triangulation is used to link the points on the second input. This ensures that a good triangle correspondence is achieved. Fig. 6 illustrates an example triangulation of a point set marked on a collection of footprints.

Some maps exhibited errors in the form of creases or discontinuities along seams where maps are combined. If the user notices such a discontinuity then they may define a line along it. This line can then be used, along with the other corresponding points, as an input to a Constrained Delaunay Triangulation [20]. This will ensure a more accurate transformation for all areas on either side of the discontinuity.
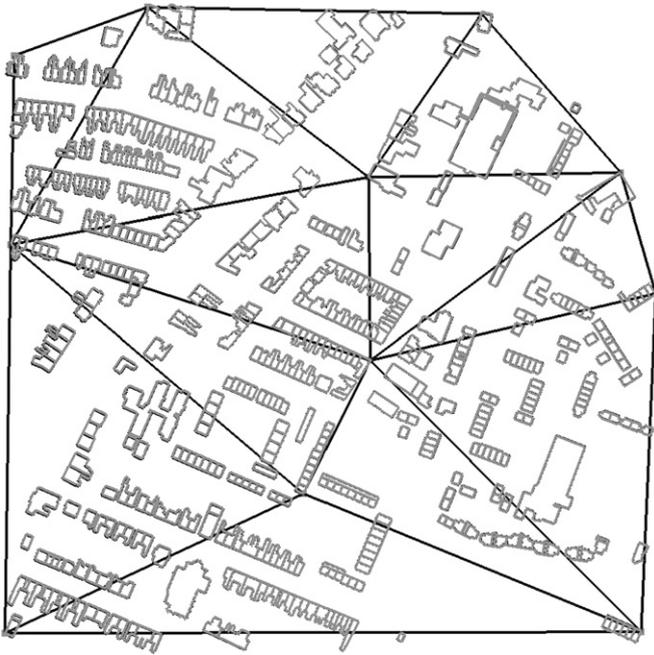
Fig. 6. An example triangulation of a point set used in the alignment process is shown. The map represents modern vector data of an area of Norwich, available from Ordnance Survey (© Crown Copyright/database right 2011. An Ordnance Survey/EDINA supplied service.).

Table 1
A table showing the errors when aligning modern vector data to an image of an archive map. The top two rows compare modern vector data to an image of the same vector data in order to illustrate the precision of aligning a map to itself. Maps M1, M2 and M3 depict different areas of Norwich, UK, in 1880 and 1885.

Map title	Piecewise linear	No. of points	Error (in pixels)		
			Min.	Max.	Avg.
Test	No	4	0.42	1.54	0.82
Test	Yes	11	0.28	0.95	0.57
1885 M1	No	4	1.12	2.89	2.03
1885 M2	No	4	1.55	3.36	2.25
1880 M3	No	4	5.39	8.54	6.44
1880 M3	Yes	9	1.24	4.21	3.04

Once the corresponding triangles have been defined, a unique per triangle pair transformation matrix can be used to align the vector data within each triangle to its corresponding building footprint on the image.

Table 1 details the errors, measured in pixels, for three example maps (M1, M2 and M3) in addition to two self-alignment tests. The self-alignment is based on aligning modern vector data to an image of the same portion of the vector data. The first two rows report very small minimum, maximum and average errors in pixel coordinates between the transformed coordinate and the original point clicked to demonstrate that the approach works satisfactorily when aligning a map to itself. The top row utilises a single affine transformation for the whole map, whilst the second row performs the alignment with a piecewise linear affine transformation based on a triangulation of several pairs of corresponding points. The bottom four rows of the table test the alignment of modern vector data to various different maps. Modern vector data may be aligned with the 1885 maps, depicting areas of Norwich, with very small errors by using a single transformation applied to the whole map. The 1880 map of central Norwich, M3, contains a discontinuity and therefore if only a single transformation is used then larger errors will occur when compared to using the set of piecewise linear

affine transformations and defining the discontinuity. This can be seen in the errors reported in the last two rows of Table 1. For example, the average error of the single transformation is approximately double that of the piecewise linear approach.

The piecewise approach may not be sufficient for very large maps where rubber sheeting techniques may be required, as utilised by MapAnalyst [4]. However, for urban environments the two alignment strategies discussed in this section have been shown to work with small errors.

4.2. Identifying differences in the data sets

A comparison process can be performed to check for potential changes to the area. To identify the differences between two data sets they must first be transformed to the same coordinate system. The transformation matrix required to transform the vectorised building footprints to the coordinates of the image of the map is found using one of the alignment strategies discussed in the previous section. Once the transformation matrix is known, each line of the vectorised footprints can be transformed accordingly. After transformation, each line is then rasterised to a pixel grid aligned to the map image allowing for a pixel-by-pixel comparison of the data. The comparison algorithm proceeds by walking along each rasterised line for a given footprint counting up the edge pixels in the map image which are within a small threshold. Depending on the accuracy of the alignment, this threshold may be adjusted to relax or focus the comparison area. A one pixel threshold results in a three-pixel-wide line. A percentage is calculated based on the number of edge pixels found over the total number of pixels tested. A user specified threshold is used to determine whether the percentage is high enough to conclude that the rasterised footprint is present in the map image being compared. An overview of this comparison procedure is described using pseudo-code in Fig. 7.

Fig. 8 shows the alignment of a portion of modern vector data (Fig. 8(b)) to an image representing the same area at a different time period (Fig. 8(a)). Once the inputs are aligned the comparison algorithm may be used to determine those buildings which are present in both maps. Fig. 8(c) illustrates the black outlines of the vector data aligned to the image. It can be seen from this figure that only a few of the buildings line up and therefore have remained unchanged, whilst significant redevelopment has taken place on a large proportion of the area. Fig. 8(d) illustrates only those black outlines which were selected by the comparison algorithm. The next section describes the performance of the extraction and comparison algorithms.

```
0 Begin Procedure: compareBuildingToImg
1     FOR each line in building
2         rasteriseLineToPixelGrid(StartPoint, EndPoint)
3         FOR each point in rasterised line
4             INCREMENT count of points tested
5             IF edge pixels found under line THEN
6                 INCREMENT number of matches
7             ENDIF
8         ENDFOR
9     ENDFOR
10    percent = number of matches / count of points tested
11    IF percent < threshold
12        return building matches
13    ELSE
14        return building does not match
15    ENDIF
16 End Procedure: compareBuildingToImg
```

Fig. 7. Comparison procedure for comparing vector data to an image and detecting changes.

5. Results

To evaluate the performance of the techniques it is necessary to compare the semi-automatic extraction process with a benchmark set of results provided via a fully manual extraction. The accuracy of the output and the time taken to obtain it were both calculated in order to establish if the methods presented are advantageous when compared to a fully manual footprint extraction. Table 2 provides details of the maps that were used, including maps of different resolutions and with varying numbers of footprints.

Fig. 9(a) and (b) shows images of two of the input maps at different time periods. Fig. 9(c) illustrates the modern vector data of the same area depicted in (a) and (b) which is used for alignment with these images.
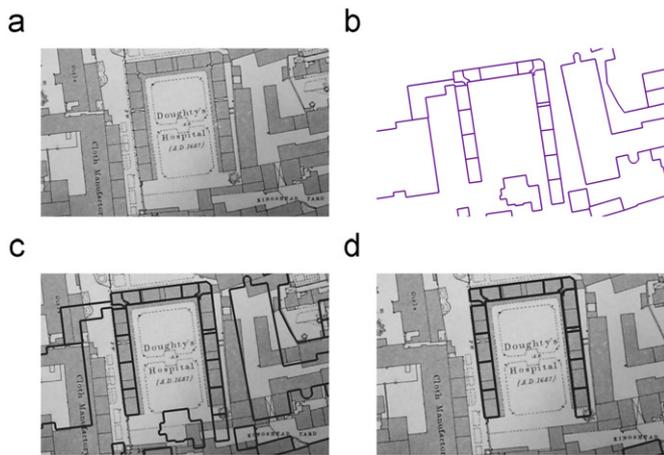

Fig. 8. Aligning vector data (obtained from Ordnance Survey, © Crown Copyright/database right 2011. An Ordnance Survey/EDINA supplied service.), shown in (b), to an image of an archive map, shown in (a). Once they have been aligned, as illustrated in (c), a comparison algorithm may determine which buildings are present in both inputs. The black outlines shown in (d) illustrate those buildings present in both.

Table 2
A table showing the properties of the maps tested. The areas cover St Andrew's Hall (SAH) and the Guildhall (GH), in Norwich, UK, at the years indicated.

Map title	No. of footprints	Width (pixels)	Height (pixels)
SAH (1885)	29	500	500
SAH (1943)	27	750	750
GH (1885)	112	1404	776
GH (1938)	104	1018	586

The times were recorded for the footprint extraction process, both for the manual approach and the semi-automatic approach. As shown in Table 3, the semi-automatic extraction is broken into three separate parts: automatic step (Auto.), manual editing that may be required (Man.) and finally the comparison (Cmp.) if the extraction is to be assisted by comparing it to another map from a different time period. If the map was extracted without the comparison step, then the column in the table contains a hyphen.

The majority of the buildings present in the tested maps could be extracted automatically, requiring a small effort afterwards to manually edit the results or add additional buildings which were missed. Despite the fact that the 1943 St Andrew's Hall (SAH) map is monochrome, the extraction algorithm was able to automatically identify the building footprints without the benefit of colour shading to assist because any white areas that do not represent buildings are generally large and are removed by one of the Floodfill algorithm's termination conditions. One map, Guildhall (GH) 1885, had poor levels of contrast in some sections of the map meaning building edges were not clearly defined. This resulted in some errors whereby several neighbouring buildings were identified as one, necessitating more manual interaction to correct errors. Repeating this test using aligned modern vector data helped to reduce this problem by improving accuracy of results, and consequently reducing the amount of manual intervention required.

To evaluate the accuracy we compare the results of the automatic extraction process with a benchmark set of results, a fully manual extraction created by a user drawing around each building on a map. This will allow us to determine error values showing the level of accuracy of the automatic process.

To compare the two sets of results, for each pair of corresponding building footprints, the footprints are rasterised to a

Table 3
A table showing the timings of the extraction process, showing automatic extraction time (Auto.), manual adjustment time (e.g. to add in buildings that were missed) (Man.), and time for comparison between modern vector data and the historic map, where applicable (Cmp.). The methods are compared to the time taken to manually extract the same area by drawing around every building footprint.

Map title	Semi-auto. Time (s)			Manual time (s)
	Auto.	Man.	Cmp.	
SAH 1885	3	60	–	360
SAH 1943	36	40	–	420
SAH 1943	13	30	0.41	420
GH 1885	21	780	–	1680
GH 1885	18	400	0.94	1680
GH 1938	16	240	–	1320

Fig. 9. An example set of data used for testing is shown. The three items depict St Andrew's Hall (SAH) and the surrounding area from 1885 (a), 1943 (b) and present day vector data (c), available from Ordnance Survey (© Crown Copyright/database right 2011. An Ordnance Survey/EDINA supplied service.).

Table 4
A table showing the accuracy of the automatic extraction process. The error values are calculated by comparing the results of the automatic extraction process with the results of a benchmark set of results, a fully manual extraction. Minimum, maximum and average error values are reported.

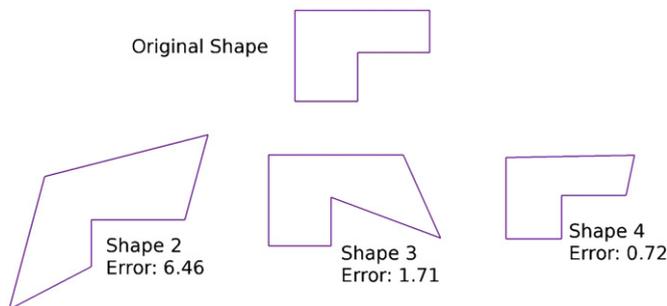| Map title	Manual vs. semi-auto.			Comparison
	Min.	Max.	Avg.	
SAH 1885	0.04	7.20	0.90	No
SAH 1943	0.03	3.82	0.52	No
SAH 1943	0.03	4.54	0.64	Yes
GH 1885	0.65	6.58	3.13	No
GH 1885	0.31	6.71	1.87	Yes
GH 1938	0.01	3.07	0.45	No

Fig. 10. The error when comparing three modified shapes to the original footprint is shown. This error is calculated using an image based error metric.

pixel grid. An image based metric may then be used to determine the distance between two feature vectors, each comprised of a set of n pixels representing the building footprints, in the form $[(x_1,y_1)\ldots(x_n,y_n)]^T$. The distance metric used is based upon central moments. Hu [21] proposed a set of seven moment invariants, which are invariant to translation, scale and rotation. The moment invariants are based upon scale-normalised centralised moments of up to order 3. The invariants are calculated and the difference between two footprints is obtained by calculating the Euclidean distance between two sets of moment invariants.

The minimum, maximum and average distances between each pair of buildings are displayed in Table 4. The distances reported here need to be put into context so that the values can be appreciated. Fig. 10 illustrates an original footprint, top, and three distorted footprints underneath. The error when comparing each of these distorted footprints to the original is reported in the image, with higher error values representing greater differences between the two shapes. If the original is compared with itself this would result in an error of zero.

One can see from Table 3 that the automatic extraction times are significantly reduced when compared to the fully manual extraction. Additionally, the average errors reported in Table 4 are mostly very small for the maps tested. However, there are some footprints which exhibit larger errors (as seen by high maximum error values), most probably caused by the poor quality of the maps which can contain blemishes and artefacts affecting the accuracy of the extraction methods in some cases. However, the comparatively low average error values reported support the suggestion that the higher maximum errors are likely to be from isolated cases. Fig. 10 shows examples of distorted shapes and their error values to serve as a reference for these values.

The use of the comparison process can be seen to aid the extraction process. For example in Table 3 the result for the 'GH 1885' map, shows a large decrease in the amount of time required for manual editing of the results. This is because the reliable modern vector data was able to be used for many of the buildings

and hence save the need for automatic extraction which could have been problematic on this image which suffered from low contrast in some areas. In Table 4 it is shown that the use of the modern data also helped to reduce the average error value for this map.

6. Conclusions

In this paper techniques have been presented to semi-automatically extract building footprints from archive maps. The resulting vectorised shapes have been evaluated by comparing them against a set of footprints manually digitised from the same map. The semi-automatic approach significantly reduces the extraction time and it is shown that the results are comparable in terms of extraction accuracy. The accuracy is achieved through the use of an edge mask to aid in constraining the footprint to the edges of the buildings on the map.

Alignment and comparison strategies have been proposed when considering the extraction of several maps of the same area at differing time periods. We have shown that the extraction times can be reduced if some buildings remain in two maps and therefore they will not need to be extracted twice. The alignment also has the additional feature of enabling maps with inaccuracies in scale to be transformed through a set of linear affine transformations so that it is faithful to modern data or previously verified vector data. In Fig. 1 we showed an example of a part of a virtual city based on the footprints extracted from an archive map. In the future one could create several versions of the same city so that the changes to an urban environment may be explored.

Future work could include further analysis of the buildings when comparing data sets. For example, historic buildings may exist in modern vector data, but have additional extensions that were not present in earlier map data. It may also be the case that large historic buildings have since been subdivided into smaller properties. Methods could be devised to recognise these instances where buildings are primarily the same but have been modified. A more complex strategy for merging neighbouring footprints could also be investigated, particularly when merging buildings of very different dimensions.

References

[1] ArcGIS. URL ⟨http://www.esri.com/software/arcgis/index.html⟩ [accessed 2010].
[2] Mueller P, Wonka P, Haegler S, Ulmer A, Gool LV. Procedural modeling of buildings. In: Proceedings of ACM SIGGRAPH 2006/ACM transactions on graphics (TOG), vol. 25 (3); 2006. p. 614–23.
[3] Laycock R, Laycock S, Day A. Reconstruction of large cultural heritage sites from archived maps. In: VAST. Malta; 2009. p. 41–8.
[4] MapAnalyst. URL ⟨http://mapanalyst.cartography.ch/⟩ [accessed 2010].
[5] Morgan M, Tempfli K. Automatic building extraction from airborne laser scanning data. In: Proceedings of 19th ISPRS congress; 2000. p. 616–23.
[6] Stumpfel J, Tchou C, Yun N, Martinez P, Hawkins T, Jones A, Emerson B, Debevec P. Digital reunification of the parthenon and its sculptures. In: VAST 2003. Brighton, UK; 2003. p. 41–50.
[7] Zhang K, Yan J, Chen S. Automatic construction of building footprints from airborne lidar data. IEEE Transactions on Geoscience and Remote Sensing 2006;44(9):2523–33.
[8] Kim Z, Huertas A, Nevatia R. Automatic description of complex buildings with multiple images. Computer Vision and Image Understanding 2004;96:60–95.

[9] Parish Y, Mueller P. Procedural modeling of cities. In: Proceedings of ACM SIGGRAPH 2001. ACM Press/ACM SIGGRAPH, ACM Press; 2001. p. 301–8.

[10] Guidi G, Frischer B, De Simone M, Cioci A, Spinetti A, Carosso L, Micoli L, Russo M, Grasso T. Virtualizing ancient Rome: 3d acquisition and modeling of a large plaster-of-Paris model of imperial Rome. Videometrics VIII 2005;5665:119–33.

[11] Fischer A, Kolbe T, Lang F, Cremers A, Förstner W, Plümer L, Steinhage V. Extracting buildings from aerial images using hierarchical aggregation in 2d and 3d. Computer Vision and Image Understanding 1998;72(2):185–203.

[12] Peng J, Zhang D, Liu Y. An improved snake model for building detection from urban aerial images. Pattern Recognition Letters 2005;26(5):587–95.

[13] Dornaika F, Hammoudi K. Extracting 3d polyhedral building models from aerial images using a featureless and direct approach. In: Machine vision applications. Japan: Yokohama; 2009.

[14] Suzuki S, Chikatsu H. Recreating the past city model of historical town kawagoe from antique map. Archives of Photogrammetry and Remote Sensing 2003;XXXIV-5/W10:1–6.

[15] Laycock R, Drinkwater D, Day A. Exploring cultural heritage sites through space and time. ACM Journal of Computing and Cultural Heritage 2008;1(2):1–15.

[16] Shimuzu E, Fuse T. Rubber sheeting of historical maps in gis and its application to landscape visualization of old-time cities: focusing on tokyo of the past. In: 8th International conference on computers in urban planning and urban management; 2003.

[17] McMaster R. The integration of simplification and smoothing algorithms in line generalization. Cartographica 1989;26(1):101–21.

[18] Rosten E, Drummond T. Machine learning for high-speed corner detection. In: European conference on computer vision, vol. 1; 2006. p. 430–43.

[19] Delaunay B. Sur la sphre vide. Otdelenie Matematicheskikh i Estestvennykh Nauk 1934;7:793–800.

[20] Chew LP. Constrained delaunay triangulations. In: Proceedings of the third annual symposium on Computational geometry, SCG '87; 1987. p. 215–22.

[21] Hu M. Visual pattern recognition by moment invariants. IEEE Transactions on Information Theory 1962;8:179–87.