Abstract
This work introduces techniques to facilitate large-scale Augmented Reality (AR) experiences in unprepared outdoor environments. We develop a shape-based object detection framework that works with limited texture and can robustly handle extreme illumination and occlusion issues. The contribution of this work is a purely geometric approach for detecting marker-like objects under difficult and realistic outdoor conditions. We demonstrate these techniques for mobile AR experiences by detecting and tracking star-shaped pentagrams embedded in the Hollywood Walk of Fame at 30Hz on a Nokia N900 phone.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Tracking

1 Introduction
The goal of this work is to deploy a large-scale mobile AR experience that exploits outdoor urban landmarks. We use the pentagram-shaped markers in Hollywood as an example that is difficult for existing methods [9]. Being on the floor, millions of people walk over them causing wear and appearance changes. Lighting variation with shadows, time of day, glare, and specular highlights pose challenges for appearance-based feature tracking. Detection and tracking in crowded urban streets must deal with extreme occlusion. Appearance-based descriptors like SIFT [8] are insufficient to describe the star shape because of its symmetry and lack of texture. With over 2000 different stars, storing templates for matching is clearly not feasible. Figure 1 shows a result of our detection method with virtual content overlays at the Hollywood Walk of Fame.

We make a number of contributions. At a high level, we are motivated by the challenge of quickly making AR markers and outdoor visual tags from existing landmarks. The preponderance of commercial logos and facility signs in urban scenes make them good candidates as tracking targets. A key design choice was to avoid pixel-based comparisons and use purely geometric methods for detection. Edge segments are extracted and robustly chained to build structural features from an image. These features called \( k \)-chains capture salient geometric structures. A novelty of our work is the direct use of these features to identify perceptual properties such as symmetry and connectedness. We introduce a hypothesize-and-test procedure that infers the object from a minimal set of \( k \)-chains based on the shape model. This makes the detection process significantly more efficient for AR than matching to a database of templates. We have built and demonstrated a star tracker using these techniques on a Nokia N900 phone.

Existing fiducial based tracking is reliant on easily thresholded images [6, 5]. Despite its simplicity and moderate robustness, the environment needs to be instrumented. Wagner [9] and Klein [7] demonstrated simplified Natural Feature Tracking (NFT) on a mobile phone for AR applications. Lack of texture and symmetric patterns can confuse these algorithms. Shape-based methods [4] that use edges alone can be sensitive to occlusion. There is a rich literature in Computer Vision on wide-baseline image matching using edge and line signatures [1]. Recently, perceptual grouping of line segments has been used successfully in object recognition and detection [3]. These are based on properties such as connectedness, convexity, parallelism, and proximity. Our work is most inspired by Ferrari et al. [3] which used groups of adjacent boundary segments for image matching. They proposed a family of scale invariant local shape features called \( k \) adjacent segments (\( k \)AS) formed by chains of \( k \) connected, roughly straight contour segments. This paper demonstrates techniques to efficiently assemble such features into model shapes for AR.

2 Image Processing
Our shape recognition algorithm relies on grouping edges in the image to form a known shape. Two complementary schemes for line segment detection are detailed below. The first method is the Burns line segment extraction algorithm [2] that runs in linear time. The detected segments are robust to lighting variations and cast shadows. Approximately 400-500 lines are detected from a typical 640 x 480 image in our dataset (Fig. 2a).

A more efficient but less stable alternative is color-based segmentation. Although sensitive to illumination and shadow effects, our shape recovery method can tolerate a large amount of noise in the initial processing stage. A 2-color Gaussian mixture model is constructed from manually labeled pixels within and outside the region of interest. The line segments are extracted from the mask by connected components grouping followed by polygon approximation. These resulting points form a polygonal chain; each consecutive set of points is then treated as an individual line segment. The next section describes the process of extracting structural features from the line segments to detect a known shape.

3 Shape Detection with Chained Edge Segments
Let \( M(x) \) be the model shape with parameters \( x \). Under perspective imaging, the shape \( S = (p_1, p_2, \ldots, p_n) \) is observed as a polygonal...
While most descriptors encode the appearance of pixels around the number of turns taken exceed the vertex of a graph. Chaining is initiated from a node until the vertex of a graph. We use a best first graph traversal that treats each line segment as a tree, checking compatibility with its neighbors. We avoid pruning or thresholding out potential lines early in the pipeline to pass on as much information as possible into the grouping stages.

Chaining two line segments is driven by local constraints. A compatibility function $V(l_i, l_j)$ for two neighboring segments is true if and only if one of the endpoints of $l_i$ passes near an endpoint of $l_j$ and the lines are directed towards each other. We define a distance metric $D(l_i, l_j)$ which returns the Manhattan distance between the two closest endpoints of $l_i$ and $l_j$ if $V$ is true and $\infty$ otherwise. $V$ can be tuned with additional knowledge of shape, but we avoid enforcing constraints too early in the pipeline. $k$ is the number of turns taken when all the line segments are chained together.

Constructing the $k$-chain is similar to path planning algorithms. We use a best first graph traversal that treats each line segment as the vertex of a graph. Chaining is initiated from a node until the number of turns taken exceed $k$. Similar to the results in [3], we have found that pairs of line segment chains with $k = 2$ makes the best compromise between discriminative ability and repeatability; low values of $k$ also keeps the detection process efficient. Therefore, all 2-chains are extracted from the image. Since these features might be composed of several segments (chaining two collinear segments will not increase $k$), we simplify them to form a descriptor of it’s three endpoints $c_1, c_2, c_3$. The middle point $c_3$ is the intersection point of the two uniquely oriented line segments in the 2-chain. While most descriptors encode the appearance of pixels around the feature, this descriptor encodes local geometric structure of the line segments in the image.

Model fitting is an iterative process and uses a hypothesize-and-test scheme. Given $N$ $k$-chains extracted from the image, each iteration randomly selects the minimum set of $k$-chains $K = k_1, k_2, \ldots, k_m$ required to determine hypothetical model parameters $x$. For objects exhibiting geometric symmetry, $m$ is likely to be low and in our experiments $N$ is typically 25-30 even from several hundred line segments. The parameters of the model are estimated from this minimal set of $k$-chains and tested for validity. If valid, the approximate model is refined using all the observed line segments extracted around the model shape. The pentagram, for example, is defined by $S$ self-intersecting lines for which $m = 3$ 2-chains would constitute a minimum set. This technique is similar to methods like RANSAC which iteratively fits a model to observed data with outliers. The combination of top-down shape knowledge and bottom-up geometric features make the detection robust. Using the star as an example, the next section describes some effective methods for fitting model shapes to $k$-chain features.

### 4 Star Detection

The pentagram or 5-pointed star is the simplest regular star polygon. It contains ten points (the five external points of the star, and the five vertices of the inner pentagon) and can be constructed by connecting alternate vertices of a pentagon. It can also be constructed by extending the 5 edges of an internal pentagon until the lines intersect. We exploit both of these properties for shape fitting. For pose, we estimate the location of the 10 corners and make correspondences with “ideal” corners from a fronto-parallel view. Due to symmetry, a unique orientation is determined by labeling the internal corner closest to the circular icon as $p_1$ and the remaining 9 corners $p_2, \ldots, p_9$ in anti-clockwise sequence.

![Figure 3: Reconstructing a star from 2-chains. Three such features with 5 unique directions are required to reconstruct the pentagram geometry. Similar to RANSAC-based feature matching, we sample from the list of $k$-chains and check for the star configuration. The thick red, green, and blue lines show a specific example of three 2-chains that adhere to the geometry. The thinner lines show the $k$-chains extended to infinity (only one direction plotted for clarity) and their intersections are plotted as squares.](image)
we compute the intersection $I_i$ of all pairs of these line. Figure 3 shows an example of three valid $k$-chains plotted in thick lines colored red, green, and blue. The colored squares are intersections of these line segments when extended to infinity; we check that they have reasonable bounds (parallel lines might stretch to infinity in which case this is not a star). Intersections are required to fall within a rectangular boundary centered on the image and twice its dimensions.

An efficient way to check for the star configuration is to first take the convex hull of intersection points in $I$. For valid sets, the convex hull should result in a pentagon connecting the 5 external corners of the star. We then verify that a putative line from the sampled 2-chains connects every other corner of the resulting pentagon. If so, the line intersections and connections adhere to the basic geometry of the star. The five lines $L_i$ that form the pentagram are constructed by connecting every alternate vertex of the convex hull. The provisional star corners $p_1^*,..,p_5^*$ can then be computed and ordered by intersecting each of the 5 lines to determine internal corners.

A final geometric verification validates that the connections do indeed form a star-shaped pentagram. We avoid thresholding on lengths and angles which are inherently brittle under non-orthogonal views. The cross ratio is an important projective invariant on lengths and angles which are inherently brittle under non-orthogonal views. The cross ratio is defined as $|bc|/|ad|$. For pentagrams, the cross ratio of these points is equal to the golden ratio $\phi \approx 1.618$ and is an important property derived from its symmetric shape. Being invariant to perspective viewpoints, our test confirms that the intersections along each of the five lines $L_i$ have the correct ratios. To tolerate noise in the sampling procedure, we threshold the ratio to be between 1.5 and 1.7. Once the 10 corners are confirmed to belong to a star, we use all line segments detected along the boundary of the shape to re-estimate $L_i$. The corners $p_1,...,p_9$ are recomputed by intersecting the refined edges.

The final step for pose estimation is to establish correspondences between the estimated corners and a template shape description. Being symmetric, the location of the circular icon is used to identify the origin. We again exploit our chaining technique described in Section 3. Line detection results in several short edge segments on the boundary of the circular arc. Given a series of line segments, we measure the likelihood of it belonging to a circle by determining the length of segments that can be chained together such that turns are taken in only one direction (clockwise or anti-clockwise). For each internal corner, we hypothesize a possible location of the icon as a circle centered on the midpoint of the line joining the internal corner and the star centroid. We then return the origin as the corner with the Maximum Likelihood estimate of containing circular arcs.

4.1 Name Recognition

There are over 2000 stars at the Walk of Fame. While shape detection is able to correctly determine camera pose from the currently viewed star, contextual information is augmented by identifying the name engraved on it. We adopted a simple template matching approach to correlate the cropped text region from the rectified star shape to a database of names. The detected star shape is first rectified and scaled to a fixed size; a 256x56 pixel region around the text is cropped. The resulting color image is projected into the Green channel which has higher contrast between the pink marble and gold metal plate. We discovered that the background and foreground text can be approximated as two Gaussian distributions; k-means was used to recover their parameters and produce a binary segmentation of text and non-text pixels. Due to the small size of our database, a template matching approach was sufficient to correctly identify the star. The template matching score is aggregated over individual letters instead of correlating the entire segmented region. Figure 4 shows augmentations performed after name recognition.

5 Experiments and Results

We test our star detection algorithm on several videos captured at the Hollywood Walk of Fame using the Nokia N900 phone. The dataset contains sequences of multiple stars taken on different days under a range of conditions. We also tested our algorithms in indoor settings on a custom-built star plaque made of polished marble. Figure 5 gives a synopsis view of star detection and pose estimation under challenging conditions. Mixed and Augmented Reality experiences in outdoor settings must be able to address such unpredictable variation in visibility, image noise, lighting, and cast shadows. Our results demonstrate how a combination of bottom-up image processing combined with top-down semantic knowledge can address these issues.

Table 1 shows quantitative results on different videos. Recognition performance is measured as the percentage of frames in which both star boundary and orientation were correctly detected. We get over 90% detection rate on most of the sequences. The Shape column shows the percentage of frames in which the star boundary was detected correctly (based on manual inspection). The Icon column shows the percentage of frames where both boundary and orientation (location of the emblem) were determined correctly. Note that these results are independently estimated for each image to measure detection performance. During on-line operation, estimation of the icon can be made robust by temporal smoothing. This significantly improves overall detection rate. Finally, among the total of 659 frames in this dataset, there were only 3 false positives.

Our ideas are built on the premise that “if a human can detect the star, the algorithm can”. This is possible by exploiting symmetry. As long as any part of the 5 lines are visible in an image, a human can infer the corners of the star. A novelty of the $k$-chain technique is that it can efficiently identify these 5 lines from several hundred line segments. The average number of line segments detected from our data was $N_l = 475$ and the average number of $k$-chains extracted was $N_k = 28$. Exhaustively testing $N_l$ choose 5 lines to test for a star shape is not tractable. When picking a minimal sample of $k$-chains for star building, an exhaustive test would require at most $N_l$ choose 3 combinations. There is a high probability of finding a good configuration early.

We bench-marked and tested our implementation on the Nokia N900 phone running the Maemo operating system. The N900 is equipped with a 600 MHz ARM Cortex-A8 CPU and 256MB of DDR RAM. Our detection algorithm was integrated into a Mixed Reality Framework (MRF) developed in-house. The framework is developed using the Qt library and enables overlaying virtual content on images streamed from the camera. Image pixels captured by the MRF are at 400x240 pixel resolution. We use binary mask
segmentation followed by polygon approximation to extract edge segments. Figure 6 shows example results of the detected shape and pose for images captured by MRF on the device. Table 1 also shows detection accuracy for two sequences captured from the N900. Our algorithm achieved a frame rate of 30 Hz on the device.

### 6 Conclusion

We have described a robust, efficient, edge-based system for detecting markers in outdoor environments. Simple features that capture structural properties of image edges are extracted from the image. They are assembled into a model shape using an efficient hypothesize-and-test framework. The algorithm achieves over 90% detection rate on challenging images while running at 30 Hz on a mobile device. Future work involves addressing issues of pose stability, efficiency, and generalization to different markers.

### References


