# Map Construction for Mosaic-Based Vehicle Position Estimation

Patrick Rowe  Alonzo Kelly

*Robotics Institute*
*Carnegie Mellon University*
*Pittsburgh, PA 15213-3890*
*email: patrick@rec.ri.cmu.edu, alonzo@rec.ri.cmu.edu*

**Abstract.** Mosaic-based vehicle position estimation is a way of tracking the motion of a vehicle using real-time imagery. A vehicle equipped with a downward-looking camera, for example, can track its motion by matching camera images with a high resolution "map" image of the floor. This paper describes our current technique for constructing such a map. The challenges of the map building process include collecting the raw map images, aligning the raw images in a common global reference frame, insuring smooth transitions between map images at map intersections, and reducing the total amount of memory required to store the map. To date, four maps have been constructed and used by four different vehicles equipped with a mosaic-based vehicle position estimation system. Results have shown this technique to be a quick, cost effective way for providing autonomous vehicles with position estimation.

## 1. Introduction

**Mosaic-based position estimation** is a technique that uses real-time imagery to track motion over a large, previously stored, high-resolution image of a known area. A traditional paper road map can be considered a low-resolution, feature-enhanced version of such an image.

Mosaic-based positioning can be used to track the motion of a vehicle. One way is to mount a downward-looking camera to the underside of the vehicle and track an image of the floor that the vehicle travels over. Like GPS, laser, or wire guidance systems, mosaic-based positioning provides absolute position fixes that are used to damp the growth of errors that occur in a primary position estimation system such as odometry.

Improvements in computer processor speed, memory capacity, and sensor technology have reached the point where a vision-based position estimation system is possible. Mosaic-based positioning offers the advantages of cheap componentry coupled with no required infrastructure, both of which serve to lower the overall installation cost of such a system in a large facility where automated guided vehicles are used, such as a factory or warehouse. Unlike "visual odometry" systems, mosaic-based positioning also offers high repeatability in vehicle positioning, which is crucial for many industrial applications.

Previous papers on the subject of mosaic-based position estimation have discussed the feasibility of such an approach [1], and have provided a description of the tracking algorithm that is used to navigate with a mosaic [2]. This paper describes the construction of the mosaic itself, hereafter referred to as a **map**. The specific issues that are addressed include imaging the surface, aligning all of the images in a common frame of reference, and reducing the amount of memory required to store the map.

## 2. Previous Work

Navigating from imagery is a basic technique in robotics [3][4], but such techniques are often concerned with the much harder problem of a three dimensional scene, usually of unknown geometry. Mosaic-based positioning deals with a two dimensional pose and assumes the scene geometry is flat, a fair assumption to make for floors.

The idea of mosaicing, or building a large image from a collection of smaller ones, has been around for some time [5][6]. Applications for automated mosaicing have included station keeping [7], video coding [8], image stabilization [9] and visualization [10]. Recently, real-time mosaicing [11] and globally consistent mosaicing techniques [12] have emerged. However, there appears to be little literature on using mosaics for vehicle position estimation and navigation. Perhaps the lack of focus on this problem has come from the practical inabilities to store enough images and to access them fast enough to be useful during vehicle motion.

## 3. Map Image Collection

A large portion of an autonomous vehicle's life is spent travelling along predefined pathways. Any deviation from these pathways is considered unacceptable, especially in areas where humans and automated machines coexist. It is also usually the case that these pathways have been defined with respect to a preexisting frame of reference such as a CAD drawing of the facility.

We are concerned with the problem of creating a map that would allow a vehicle equipped with a mosaic-based positioning system to travel along these pathways. Since the vehicle is not intended to deviate from the pathways, we only wish to map the portions of the floor that the vehicle's camera would see.

Figure 1: The mapping rig consists of a downward-looking camera system, to acquire the images, and an odometry positioning system, to determine a relative position of each image.

### *3.1 Mapping Rig*

The first step in creating a map is collecting the images of the floor that will be traveled over by the vehicle. Associated with each collected image is a 2D pose, with the center of the first image being defined as the origin, and all other image poses defined relative to it.

As a matter of convenience, image collection is done with a special piece of equipment called a **mapping rig**, which is shown in Figure 1. The mapping rig is a heavy push cart that has a camera mounted on its underside looking down at the floor. LED's are used as a lighting source to illuminate the floor, and the underside of the cart is shielded to keep out ambi-

ent light.

Images are captured at a rate of 30 Hz. With our current mapping rig design, the area of the floor that is captured by a single image is 0.5 m by 0.665 m. These map images are twice as wide as the images that are captured with our existing mosaic-based vehicle positioning systems. The larger map size offers some flexibility by not requiring the map to fall exactly over the vehicle's path, and better ensures that the autonomous vehicle's camera image will be entirely in the map image.

Odometry encoders that ride on the large fixed wheels of the mapping rig cart provide the pose information for each captured image. The differential motion of the center of the fixed wheel axle is computed from the encoder readings and then integrated to determine the absolute pose of the mapping rig. The precise position and orientation of the camera with respect to the center of the fixed wheel axle has been measured and is used to calculate the pose of the center of each captured image. The encoders are read at a rate of approximately 100 Hz.

The mapping rig also contains a computer with a frame grabber and encoder interface card. Custom software collects the images and calculates their pose. On-board rechargeable batteries and an AC inverter provide power to the computer, lights, and camera.

To collect the images for the map, the mapping rig is pushed along the pathways that the vehicle will travel. A set of images collected during one run of the mapping rig is known as a **map segment**.

### *3.2 Mapping Preparation*

The are a few preparatory steps that need to be done before image collection. First, the area of the floor that will be mapped must appear in the same condition as it would during normal vehicle operation. This means it should be clean of any dirt, dust, tape or other debris that a standard floor cleaner would remove.

Next, the pathway locations need to be marked in some manner so the mapping rig operator knows where to go. Grooves in the floor, chalk lines, masking tape, and arced "curbs" have been used as guides during the image collection process.

A third optional pre-mapping step that aids in the off-line map construction stage is placing small markers at known locations in the pathways. For example, if a CAD layout of the building were at hand, then certain points could be measured from known features in the CAD layout, and their precise coordinates in the building's frame of reference would be known. In practice, we have used small (1 $cm^2$) pieces of reflective tape for these markings. Ideally, they should be placed at the intersections of all the pathways. The advantages of this step will become evident in Section 4.1.

### *3.3 Mapping*

Once the mapping preparation work has been done, the mapping process itself goes very quickly. The time it takes to collect the images is a function of the total length of the map segments, the number of map segments, and the distances from the end of one map segment to the beginning of the next one. The mapping rig is pushed slightly below normal human walking speed (approximately 0.5 m/s). We estimate that the start of a mapping pass at a new map segment requires about a minute of set up time. The amount of time it takes to move between map segments depends on the layout of the map itself.

Each pathway is collected as a separate map segment. We prefer to make maps with straight map segments, as this reduces the errors in the dead-reckoned image poses. However, we have successfully built maps that contain large arcs of 12 and 20 feet in radius.

## 4. Map Construction

Once the individual map segments have been collected, the next step in the map building process is placing each segment within a global frame of reference to create the complete map. One item of importance is to make sure that the images at pathway intersections, where two map segments overlap, are well enough aligned so that when tracking the map, a smooth transition is made from one map segment to the next, and the mosaic tracking algorithm does not lose visual lock.

Each map segment, which originally consists of many small overlapping images, is then condensed into one large image with a single pose, which greatly reduces the total memory required for map storage.

### *4.1 Map Construction Algorithm*

A map can be considered a graph-like structure whose edges are the map segments and whose nodes are the intersections between map segments. The map is created by placing each edge, or map segment, in the right place in the graph one by one. The images in each map segment are repositioned with respect to a global frame of reference. This is done based on distinct features in certain images that have known coordinates. We refer to these features as **lockdown points**, because they "lock down" each map segment at known locations in the global frame of reference.

It should now become clear why the pre-mapping step of placing small pieces of tape at known locations is useful. The tape marks serve as the lockdown points. Furthermore, if they occur at pathway intersections, it better ensures that the overlapping map segment images are aligned properly.

Software has been developed to allow a user to view an image that contains a lockdown point, select the lockdown point's location, and assign the lockdown point to its known global coordinates. When each lockdown point is selected, the following algorithm and corresponding functions determine how to reposition the map segment images.

*Given a map segment that contains* n *images,*
*For each image* i *in the map segment that contains a lockdown point,*
*Traverse the map segment backwards from image* i *towards image 0.*
*If an image in the map segment which already contains a lockdown point is found (call it image* j*),*
***Smooth** the images from image* i *to image* j.
*Else if the end of the map segment (image 0) is reached,*
***Rigidly move** the images from image* i *to image 0.*
*End if.*

*Now traverse the map segment forward from image* i *towards image* n-*1. Use the same rationale as above to determine if the images need to be smoothed or rigidly moved.*

#### *4.1.1 Rigidly moving map images*

Recall that when a map segment is initially collected by the mapping rig, the center of the first image is defined to be the origin of the map segment. When a lockdown point is first identified, its coordinates are also relative to this map segment origin. Given the coordinates of the same lockdown point in the global reference frame, and assuming the images are rig-

idly attached to one another, we would like to find the new global positions of all of the images in the map segment.

Let us define the map segment frame as *M*, the global world frame as *W*, the lockdown point pose as *LD*, and the image pose as *I*. Some simple homogeneous transformation operations reveal

$$ {}^{W}_{I}T = {}^{W}_{LD}T({}^{M}_{LD}T)^{-1}{}^{M}_{I}T $$

where ${}^{M}_{LD}T$, ${}^{W}_{LD}T$, ${}^{M}_{I}T$, and ${}^{W}_{I}T$ represent the lockdown point's coordinates in the map segment frame, the lockdown point's coordinates in the global frame, the image's coordinates in the map segment frame and the new image coordinates in the global frame respectively. The first two terms of the right hand side of the equation can be combined into a single matrix operator that is independent of the image poses. This operator is then used to compute the new global pose of each image in the map segment.

### *4.1.2 Smoothing map images*

The concept behind the smoothing algorithm is shown in Figure 2. Consider a portion of a map segment between two images that contain lockdown points. In this particular example, there are two images between the outer lockdown-point images. The lockdown point on the left has already been fixed in its proper coordinates with the **rigidly move** operation. The lockdown point on the right is not at its desired location. Therefore its pose, and the poses of all of the images between it and the fixed lockdown point, must be repositioned.

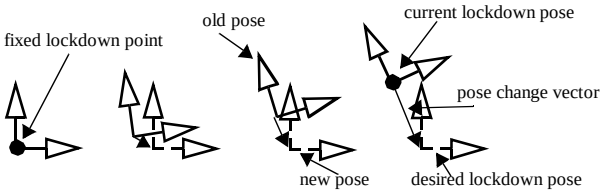


Figure 2: Concept behind the image smoothing algorithm.

The error in position and orientation of each image is apportioned linearly as a function of the distance between the two endpoint images. Thus, if an image lies exactly halfway between the two, its pose will be corrected by one half of the total error.

The smoothing algorithm is a two step process. First, the amount of angular rotation $\Delta\theta$ for each image *i* is determined by the following equation

$$ \Delta\theta_i = \frac{s_i}{S}\theta_T $$

where $s_i$ is the path length distance from image *i* to the image that contains the fixed lockdown point, *S* is the total path length between the two lockdown-point images, and $\theta_T$ is the total angular change. The path length *S* is computed by summing the distances between the centers of neighboring images in the map segment, and $s_i$ is computed in the same way up to image *i*.

The images are rotated beginning with the fixed lockdown point image and moving towards the one containing the other lockdown point. It is assumed that the images in the map segment act as a rigid body. Because of the coupling between rotation and translation,

as each image is rotated, the remaining images between it and the final image are repositioned as if they were rigidly attached. The same derivation as was done in Section 4.1.1 can be applied to find the transformation operator here.

After this step, all of the images have the proper orientation, but they may still be positioned incorrectly. The new error in x and y between the current and desired lockdown point's coordinates is found, and the change in x and y for each image is computed in the same way as the change in angular rotation. This translational change is simply added to each image pose.

### *4.2 Global vs. Relative Lockdown Points*

It is also possible to construct a map without having global lockdown point coordinates. Instead, easily identifiable image features can be selected as lockdown points as the map is constructed and assigned with their current coordinates. These "relative" lockdown points would need to occur at the intersections of the map segments, so one map segment can be attached to another.

Relative lockdown points have the advantage that the coordinates of known locations along the pathways do not need to be found, which is a time consuming process if there are many map segments. However, we prefer to use global lockdown points as much as possible for several reasons. For one, it automatically aligns the map with an external frame of reference. This is important if vehicles are commanded to navigate to preexisting pick-up or drop-off points. Using global lockdown points is also advantageous in that it removes accumulated odometry errors in the image poses for each individual map segment. In the case of relative lockdown points, images are only smoothed if the map segment closes a loop in the map. In this case, the last map segment would "absorb" all of the error for every map segment up to this point, possibly resulting in an extremely warped map segment.

### *4.3 Map Construction Example*

Figure 3 shows an example of how lockdown points are used to reposition the images in a map segment. Shown in the figure is a rather long map segment that begins in the upper left hand corner, continues south around the bottom loop, and then north around the upper loop.

Figure 3a shows the initial map segment where the positions of the map images are first determined from the raw odometry of the mapping rig. The black x's show the location of the global lockdown points. As more lockdown points are added, the map segment is brought into its proper alignment. Figure 3d shows the final aligned map, which successfully seals the loops in the map.

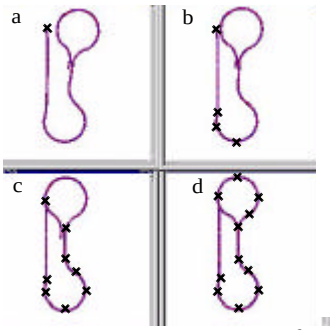


Figure 3: Four images of a map segment during the off-line map construction phase.

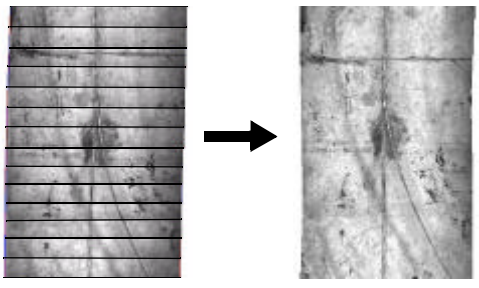

Figure 4: Map segment portion before and after image condensing.

## 5. Map Image Condensing

Once all of the map segments have been positioned at their final locations, each map segment is condensed into one large image. Figure 4 shows a small portion of a map segment before and after this process. We have found that this step reduces the required memory for storing the map by a factor of 3 to 4 depending on the amount of original image overlap. After image condensation, one kilometer of mapped pathways can be stored in approximately 120 Mbytes of memory.

The image condensing algorithm is fairly straightforward. The size of the final large image is taken to be the bounding box of the map segment. The position of the large image is defined to be the center of the bounding box, and its orientation is the average orientation of the original map segment images.

The coordinate transformation operator is then found between the large image frame and the original map segment frame. This operator determines which pixel in the map segment is to be used for each pixel of the large image. If several map segment pixels map to the same large image pixel, then the map segment pixel that is closest to the center of its image is selected. The reason for this is the center of the images offer better lighting and less distortion than the outer edges of the image.

One concern with this concept is that in creating a large image with pieces of the smaller images, artificial intensity edges may be introduced if the original images do not perfectly overlap due to either odometry error or altered poses from the smoothing algorithm, or if the lighting over each individual map segment image is non-uniform. We have not found this to be the case, however, as the odometry error that is accrued in the distance between map segment images, or the amount that an image is moved relative to its neighbor during smoothing, is much smaller than the physical area covered by a pixel. The large images appear smooth and edge free.

Because the straight map segments are never perfectly straight, or in the case of arced map segments, some of the large condensed image will contain invalid pixels. Rather than cropping the larger image so that each pixel is valid, an additional bitmap is created that records whether a pixel is valid or not. This bitmap is used by the mosaic-based positioning system to determine which portion of its image falls in a valid section of the map image. Each bitmap requires 1/8 the memory of each large image.

## 6. Results

To date, four different maps have been constructed in three different facilities with this technique. These maps have ranged in size from 80 to 200 linear meters, some containing large arcs, and all with closed loops. Four different autonomous vehicles, including forklifts, tuggers, unit-loads, and tow motors, equipped with a mosaic-based position estimation system have successfully navigated with these maps with excellent results. Total map creation, from the initial preparatory steps to the image collection to the final off-line creation and condensing, can be done in less than a day.

## 7. Conclusions and Future Work

In order to use the technique of mosaic-based position estimation, a required component is a map of the surface that will be tracked. This paper has presented our current map making

procedure, including image collection, image alignment with an external frame of reference, and map storage.

The most important enabling factor of mosaic-based positioning is there must be an appropriate amount of visual texture present in the map. This is a fundamental requirement for any correlation-based computer vision system. We have found that the majority of existing floors do exhibit enough visual texture for mosaic-based navigation to work. For example, concrete and wood floors provide excellent natural texture. Floors that have been painted a single color, however, are inadequate for mosaic-based positioning. In such cases, we have applied our own painted texture using either special textured paint rollers or artists' sponges.

One area of future research is the ability to map larger areas other than "one-dimensional" pathways. This may be required if vehicles are permitted to roam freely. Another topic for future research is combining image correlation with the odometry information to both better align map segments at pathway intersections, and to better align the images themselves within a map segment. This may help to remove odometry errors caused by wheel slip or non-flat floors. Of course, as more maps are created at various facilities, we are constantly improving our techniques for constructing the maps to make map installation as quick and automatic a procedure as possible.

## 8. References


[1] A. Kelly, "Contemporary Feasibility of Image Mosaic Based Vehicle Position Estimation," *Proceedings of IASTED International Conference on Robotics and Applications*, Santa Barbara, October 1999.

[2] A. Kelly, "A Pose Tracking Approach for Image Mosaic Based Vehicle Position Estimation," submitted to *IEEE International Conference on Robotics and Automation*, San Francisco, 2000.

[3] S. Atiya and G.D. Hager, "Real-Time Vision Based Robot Localization," *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 6, pp. 785-800, 1993.

[4] B.K.P. Horn and E.J. Weldon, "Direct Methods for Recovering Motion," *Int. J. Computer Vision*, Vol. 1, pp. 51-76, 1988.

[5] P.J. Burt and E.H. Adelson, "A Multi resolution Spline with Application to Image Mosaics," ACM Transaction on Graphics, Vol. 2, pp. 217-236, 1983.

[6] Q. Zheng and R. Chellappa, "A computational vision approach to image registration," *IEEE Transactions on Image Processing*, 2(3):311-325, July 1993.

[7] R.L. Sparks, S.M. Rock, and M.J. Lee, "Real-Time Video Mosaicing of the Ocean Floor," *IEEE Journal of Oceanic Engineering*, Vol. 20, No. 3, July 1995.

[8] M. Irani, P. Anadan, and H. Hsu, "Mosaic based representations of video sequences and their applications," *Proc. Intl. Conf. on Computer Vision*, pp. 605-611, 1995.

[9] L. Wixon, J. Eledath, M. Hansen, R. Mandelbaum, and D. Mishra, "Alignment for Precise Camera Fixation and Aim," *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998.

[10] R. Szeliski, "Image mosaicing for tele-reality applications," *IEEE Workshop on Applications of Computer Vision*, pp. 44-53, 1994.

[11] H.S. Sawhney, R. Kumar, G. Gendel, J. Bergen, D. Dixon, and V. Paragano, "VideoBrush: Experiences with Consumer Video Mosaicing," *Fourth IEEE Workshop on Applications of Computer Vision (WACV)*, October 1998.

[12] H.S. Sawhney, S. Hsu, and R. Kumar, "Robust video mosaicing through topology inference and local global alignment," *Proc. European Conf. on Computer Vision (ECCV)*, Vol. 2, pp. 103-119, Freiburg, Germany, June 1998.