Mobile Robot Localization from Large Scale Appearance Mosaics

Alonzo Kelly

CMU-RI-TR-00-21

The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

Dec 1, 2000

©2000 Carnegie Mellon University

Abstract

A new practical, high-performance mobile robot localization technique is described which is motivated by the fact that many man-made environments contain substantially flat, visually textured surfaces of persistent appearance. While the tracking of image regions is much studied in computer vision, appearance is still a largely unexploited localization resource in commercially relevant guidance applications. We show how prior appearance models can be used to enable highly repeatable mobile robot guidance that, unlike commercial alternatives, is both infrastructure-free and free-ranging. Very large scale mosaics are constructed and used to localize a mobile robot operating in the modeled environment. Straightforward techniques from vision-based localization and mosaicking are used to produce a field-relevant AGV guidance system based only on vision and odometry. The feasibility, design, implementation, and pre-commercial field qualification of such a guidance system are described.

Table Of Contents

1. Introduction - Mosaic-Based Localization

Imagine yourself flying over a city in a small airplane. Let the airplane be restricted to level flight and let the terrain below be assumed to be essentially flat. That is, let the terrain undulations be small relative to the aircraft altitude. You can see the ground below through a small viewfinder in the floor. You have a map of the city in the form of a large high resolution photograph. Your task is to locate yourself, to the nearest building, by matching the views in the viewfinder to the mosaic.

This scenario illustrates the technique of *mosaic-based localization* described in this article. Replace the viewfinder with a camera; replace the airplane with any vehicle travelling parallel to a mostly flat surface; restrict vehicle motion to the streets; and you have the general idea. This approach to localization has shown itself to be both robust and of high performance in the environments to which it is targeted.

1.1 Application to Textured, Simply Structured Scenes

This article will apply the above technique to the application of (robot) vehicle localization. We use a mosaic as the large high resolution image that is used as the map, and hence the technique will be called *mosaic-based localization*. We will restrict our attention to scenes which are locally flat enough that the use of a mosaic as a prior scene model makes sense. Such scenes will be called *simply-structured*. The work described here should apply with little modification to all such scenes when the more fundamental requirements of visual tracking are also satisfied.

The goal of the article is to introduce a relatively simple and robust solution to an important localization problem. We will exploit all available assumptions and engineering simplifications to their fullest potential if their overall impact on reliability is considered a positive one. As a result, the vision problem is so reduced in complexity that a guidance system capable of tracking velocities exceeding 1000 times its resolution per second is produced.

Using mosaics of the floor, we convert the problem of vehicle localization into a simple instance of the camera pose recovery problem of computer vision and thereby introduce a field-relevant form of vision-based localization.

Although there are clear alternatives, we will exploit the particular advantages of using floor imagery rather than images of other surfaces. Also, while many other applications satisfy our scene constraints, we will discuss the details of an application to industrial AGVs.

1.2 Description of the Approach

We will represent the environment as an appearance model - in all its photorealistic richness. While there are other ways to acquire real textures, we use mosaicking techniques to construct this model. Our technique differs from *visual odometry* [8][34] in that considerable effort is expended to create a globally consistent model. It differs from *landmark-based localization* in that the scene is represented in an iconic form rather than as a list of landmark locations.

The steps of our mosaic-based approach to localization are:

- Construct a mosaic of an appropriate area.
- Render it globally consistent and store it in persistent memory.
- Subsequently track motion over the mosaic using a visual tracker which computes camera pose.

1.2.1 Rationale

Given that a mosaic scene model can be constructed in principle, it still remains to explain why it is even worth such effort. For our purposes, a mosaic is a particularly convenient and appropriate form of prior scene model. This conclusion can be rationalized as follows:

- *Prior Models Enable Higher Tracking Velocities*: We will create and track position in persistent models rather than tracking regions of pixels from frame to frame. When speed exceeds levels at which successive images overlap in the scene, there is no information that can be tracked from image to image. However, referencing a prior model eliminates the image overlap constraint so long as some part of the model remains in view.
- *Global Consistency Imparts Repeatability:* Prior models imply that a definitive location is stored with (or repeatably derived from) the persistent model. If the model is also globally consistent, reported position becomes a one-to-one function of location and the result becomes neither time nor path dependent. When such models are used, the system becomes as repeatable as its fundamental resolution (while its accuracy depends on the quality of model calibration).
- *Iconic Models are Best in Featureless Scenes*: If features are dense in the target environment and of high enough resolution to be representable as discrete, a simplified feature-based modeling approach can be best. However, if features are rare, spatially distributed and/or subtle, an iconic representation encodes the maximum useful information in terms of providing the best immunity from false correspondence matches and highest spatial (sub-pixel) resolution.
- *Geometry Assumptions Simplify Processing*: Of course, when scene geometry can be regarded as known, algorithms need not recover shape as well as motion, and the distortion of iconic features due to motion can be predicted.
- *Image Registration Can Generate the Necessary High Resolution Appearance Models*: The ratio of excursion to required resolution for many applications will exceed the spatial resolution achievable in a single image. However, image registration can be used to generate an image whose memory footprint is limited only by the computer used.

Hence, a class of problems exists for which repeatable, high resolution, high speed localization is required and image registration can be used to produce the necessary prior model (a mosaic) for a vision-based solution.

1.2.2 Key Assumptions

When cameras are used for vision-based localization, the ability to render a scene permits navigation from real-time imagery [38]. While it is certainly possible to compute unrestricted 3D camera motion in an (even unknown) 3D scene, our application will make and exploit several more simplifying assumptions:

- Persistent Appearance: While a persistent shape assumption is commonly made of scene geometry, the use of a persistently stored model of scene appearance assumes that the actual appearance of the scene will not change significantly over operationally significant periods of time. Exceptions to this assumption are common, but the appearance change needs to be significant and it needs to occur everywhere in order to render the present technique inoperable. The process of learning slowly changing appearance is also a theoretical possibility.
- 2D Scene: We will use appearance models constructed from real imagery. While completely general 3D polygon models are certainly possible, we will assume that the scene can be represented by a mosaic mapped onto a 2D surface. This assumption applies, at least locally, to most man-made indoor and outdoor environments.

- *Substantially Flat Scene:* While the assumption can be completely relaxed in general (e.g. in computer graphics), we will assume that the scene is flat enough that self occlusion and depth discontinuities cannot occur. This assumption also applies, at least locally, to most man-made indoor and outdoor environments. Such environments often consist of flat surfaces punctuated by line intersections with other flat surfaces.
- *Restricted Camera Motion:* While arbitrary camera motion is computable, we will restrict motion to be consistent with being mounted under a terrain-following vehicle. That is, the camera moves in a plane parallel to the flat scene while being oriented parallel to the terrain normal as shown in Figure 1:. Under these conditions, the general problem of "rendering" the scene is reduced literally to that of extracting the pixels in the rectangular region predicted to be in view.
- Restricted Mosaic Topology: While not necessary in general, we will confine our attention to environments where vehicle motion is restricted to roadways or guidepaths, rather than regions wider than an image in more than one direction, except at intersections. To do so simplifies considerably the problem of constructing globally consistent mosaics. Confinement to guidepaths is required for safety reasons in the AGV application.
- Primary Position Estimate: Since we will confine the application to that of vehicles, it is useful and not overly restrictive to assume the availability of an independent estimate of camera motion between frames. This primary position estimate can be used to increase reliability and very significantly increase tracking performance and therefore vehicle speed.

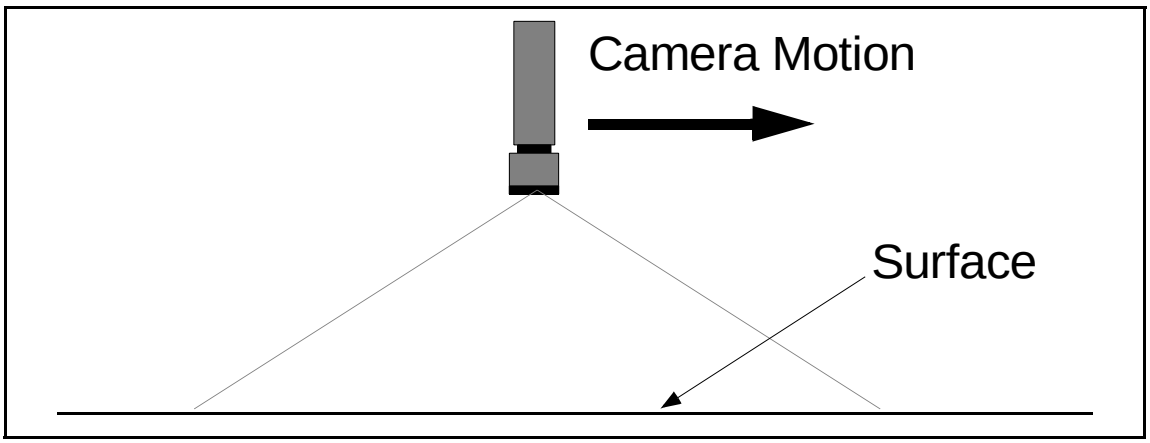

Figure 1: Simplest Scenario. Here a camera is mounted normal and at constant height with respect to a surface and it moves parallel to the surface. Assumptions imply that variations in foreshortening over the image do not occur.

1.3 Prior Work

1.3.1 Prior Localization Work

Many different techniques have been proposed for localization in general [3] and for vision-based localization in particular [43]. Certainly, navigating from imagery is a basic technique in robotics [2] [20] but such techniques often deal with the much harder problem of a three dimensional scene, often of unknown, sometimes nonrigid geometry, and often with unrestricted camera motion. While there are no doubt many ways to categorize vision-based approaches, they can be distinguished for our purposes based on:

- *Sensor Modality*: Whether they use appearance (cameras [1]) or shape (radar [12], sonar [10] or lidar [13]), or both [33]. This choice can be based on the relative visual or geometric richness of the environment.
- *Feature Detail:* Whether topological [26], spatially discrete [20][29], or extended features such as lines [2][25], or full iconic representations of the scene [14], entire images [35], or iconic features are used and manipulated.
- *Model Explicitness:* Whether a model of the environment is created or more direct sensor-based methods [11] are used to directly associate position with sensor readings - bypassing an explicit model entirely.
- *Model Determinism:* Whether deterministically or stochastically based [4] sensor predictions are generated.
- *Storage Persistence:* Whether the explicit model is stored in persistent storage or whether features are simply tracked from frame to frame [30].
- *Global Consistency:* Whether the model is globally consistent [7] or it consists of a series of locally consistent sub-models that have not been registered.

Other important distinctions include the information content of the map or model used, the algorithm used to predict position, whether the map is being learned or used or both [18], the structuredness of the environment, and whether a 2D or 3D pose is generated.

1.3.2 Prior Mosaicking Work

The practice of *image mosaicking*, of producing larger images from the union of smaller ones registered in their regions of overlap, is an old one. Mosaicking is closely related to image registration and rectification, and camera motion analysis. The technique of construction of mosaics by computer is also relatively old [6].

Automated mosaicking [51] is often useful in its own right. Applications include station keeping [41], video coding [23], image stabilization [50], and visualization [42]. Only recently have near real-time [37] and globally consistent [36] mosaicking solutions emerged.

1.3.3 Prior Motion Estimation Work

The literature on determining the motion of a camera and/or the geometry of a scene is extensive. Motion can be recovered from a known scene [48] and this problem is related to visual odometry. Scene structure can be determined from camera motion [45][32]. Shape and motion can also be determined simultaneously [44] and all shape and motion assumptions seem ultimately unnecessary.

1.3.4 Prior Visual Tracking Work

Once a camera is permitted to move relative to a scene, one can observe correspondence or flow [40]. For correspondence, the related problem of visual tracking [19][39] becomes important.

1.3.5 Prior Pose Refinement Work

Of course, the distinction between a rigid scene and an object is largely unimportant and the distinction between location or motion of the camera versus that of the scene is unimportant. It is well-known that relatively few correspondences between the image and the scene are necessary to constrain the relative pose of a camera and a known object or scene [22]. Fairly general 3D solutions for finding the relative pose have been known for some time[16].

1.3.6 Prior Image-Based Rendering Work

Since we will render predicted imagery, this work is related to image-based rendering. The problem of image-based rendering [9][27][49] is related to visual tracking in that the motions and deformations of all regions of the image are being predicted. Solutions enable the prediction of imagery in all its photorealistic richness from camera views that have not necessarily been previously recorded, but are close enough to views which have been recorded to enable view interpolation.

1.4 Characterization of Vision Problem

Much of this literature solves problems far more difficult than our problem here. We will be interested in computing the pose of a camera mechanically constrained to move in 2D over a known, rigid, flat, unambiguously textured scene. Our problem is a trivial version of the camera pose recovery problem. It is not even as difficult as camera calibration since we will rectify imagery to exhibit ideal pinhole geometry. There are only three degrees of freedom of motion and two corresponding points will suffice to locate the camera relative to the mosaic. Further, an external estimate of camera motion will be exploited to heavily constrain the search for features.

Feature matching will be reduced to trivial rigid template matching since depth is everywhere constant and we even have complete control over the lighting. We will find it necessary to "render" regions of the predicted scene from the assumed camera location, but our geometric constraints literally reduce this problem to extraction of a rectangular region from another image.

1.5 Motivation - Application to Vehicle Localization

The primary reasons for investigating this problem of mosaic-based localization - of localizing a camera that moves over a scene modelled by a mosaic - is that it applies to an important application because its underlying assumptions are usually valid and its strengths make it competitive to alternatives.

1.5.1 Important Problem - Vehicle Localization in Simply Structured Environments

The camera under discussion can be affixed to a vehicle whose location is of primary interest. In this case, camera localization generates vehicle localization. Our scene geometry assumptions (flat surfaces) apply to this problem in the case of man-made environments - both indoor and outdoor.

In particular, a robot vehicle introduced into such environments normally moves over a mostly flat floor surface. Walls and ceilings, if present, are also composed of locally flat surfaces punctuated by occasional abrupt shape discontinuities. It is often the case that at least some of these surfaces present at least some areas of visual texture.

1.5.2 Automated Guided Vehicles

In many cases, such vehicles operating in such environments are known as automated guided vehicles (AGV's) and these are perhaps the most commercially relevant mobile robots today. A typical such vehicle is indicated in the figure below:

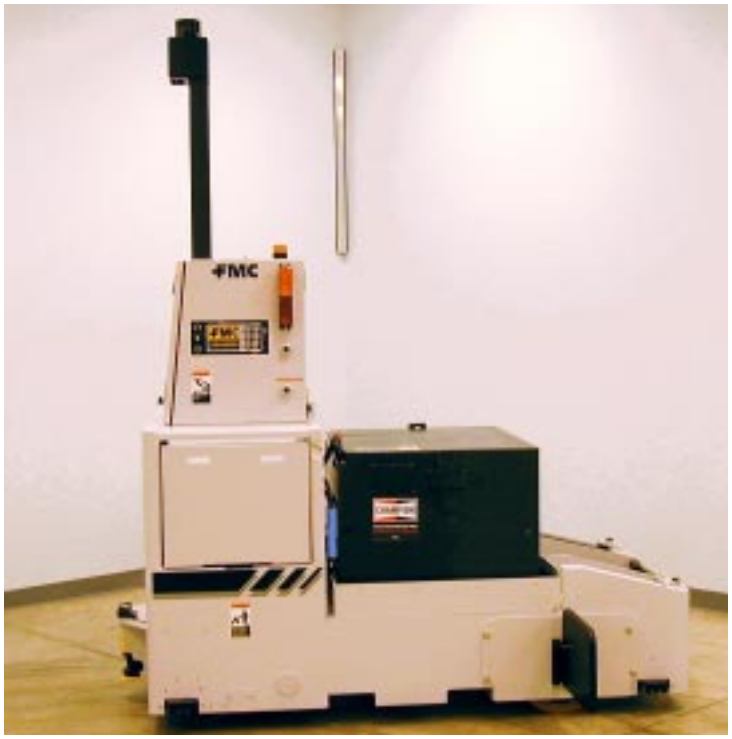

Figure 2: Industrial AGV. Such vehicles are the most commercially relevant mobile robots today comprising a global market of $566M in 1998. This particular vehicle is a tug AGV designed for material handling applications.

This particular one is a tug-AGV designed for material handling applications. Typically, several such vehicles operate simultaneously in a given area. The architecture for their control is highly centralized and based on periodic radio communication between each vehicle and a central traffic manager.

Obstacle avoidance is often limited to mechanical bumpers, although lasers are becoming more important. They are *optimistic*, operating under the assumption that the path ahead is clear in the absence of evidence to the contrary.

AGV's do not normally use any "models" of the geometry of their environment encoding where the walls, shelves, or doors are. Even the roadways or "guidepaths" to which their motions are typically confined, and the locations of special places where loads are to be picked or placed, are known only to the central traffic control computer.

Such vehicles are the mobile equivalent of the pick-and-place manipulator. The commercially optimal configuration seems to be to keep such vehicles almost entirely blind and have them rely on at least two major assumptions:

- *Repeatable Position Estimation:* ensures that the vehicles remain on the paths that were laid out at installation time to be free of potential collision with the structure or infrastructure of the site.
- *Enforced Environmental Structure:* the guidepaths are not only *supposed* to be clear of obstacles but there is at least a loose policy of keeping them free of obstacles.

1.5.3 Service Robots

Another commercially relevant application where scenes are simply structured scenes is service robots. Such vehicles are (or could be) used in hospitals, warehouses, hotels, museums, subways, etc. for such diverse purposes as floor care, mail delivery, light material handling, tour guiding,

entertainment, and surveillance.

1.5.4 Commercial Promise

In summary, typical conditions in plants, office buildings, and yards present a new opportunity to produce inexpensive and highly repeatable position estimation based on template matching of textured simply-structured scenes.

Mosaic-based localization is ideally suited to such environments because it assumptions are valid in the environments in which they currently operate, and its repeatability is a necessity (as the architectures of vehicle systems are currently formulated).

Commercial alternatives for the positioning of vehicles in yards and manufacturing facilities include:

- *wire guidance:* where crosstrack error is measured from inductive pickoffs sensing wires that are literally embedded in the floor of a building.
- *laser guidance:* where a spinning laser beam senses the bearings and sometimes the ranges of retroreflective fiducials mounted on the walls of buildings.
- *inertial guidance:* where dead reckoning from gyroscopes and accelerometers is augmented by occasional position fixes from special fiducials in the floor.
- *radio guidance*: forms of indoor GPS are reported to be under development in industrial laboratories.

Mosaic-based positioning promises to compete favorably with the above alternatives in suitable environments because:

- No infrastructure such as wires or reflective beacons is required. This makes the system harder to sabotage and easier to install.
- Resolution is limited only by optics. It can, in fact, be used on microscopes, cameras, and telescopes.
- Capital cost is limited to that of a camera, lighting, off-line storage, and a capable processor - which may be required for other reasons.
- Installation cost is limited to the labor and time required to map the environment by driving over all necessary guidepaths or areas only once.

2. Feasibility Analyses

This section will cover some of the more obvious questions which immediately arise when the concept of tracking a large scale mosaic is first considered. Some of the requirements that such a guidance system must satisfy in order to function effectively are discussed, and then analyses are offered which demonstrate that these requirements can be met under certain assumptions.

The algorithms and results described in this section constitute a feasibility analysis to verify the general concept. Many are not part of the real-time implementation.

2.1 Requirements

Mosaic-based localization has the same scene texture requirements of all correspondence-based vision algorithms. The scene must exhibit texture which is:

- strong enough,
- common enough,
- sufficiently unambiguous.

Of course, processing capability commensurate with the motion being tracked is also important in visual tracking. In our AGV application, the need to track feature velocities of 1000 pixels per second raises such concerns immediately.

When the notion of a persistently stored mosaic is considered, some immediate issues are:

- Sufficient image storage and fast enough access is needed.
- The assumption of persistent appearance over the long term is usually challenged.
- Constructing globally consistent mosaics can be a difficult process.

With the exception of the last point, this list of issues can be reduced to texture, memory and processor speed. Systematic consideration of these potential problems leads to the conclusion that contemporary sensing and computing technology are adequate to the task in suitable environments. Following sections discuss these matters in more detail.

2.2 Storage

When tracking against an entire mosaic, it is necessary to have the entire region that could be traversed available at least in off-line memory, if not in RAM. Estimates of storage requirements come down to a determination of the surface area that must be covered and of the required pixel size on that surface.

2.2.1 Area

Area is straightforward to estimate in principle. The union of all the floor area that will ever be seen by the camera is needed. In our application, the guidepaths are fixed, well-studied, and available well before vehicle installation.

2.2.2 Resolution

Resolution is determined by the worst case requirement generated from several concerns:

- *texture scale*: If the surface texture exists only at fine levels of detail, the pixels must be correspondingly small to resolve it. Most surfaces we have encountered, however, exhibit texture at all practically useful scales.

- *pose resolution:* The resolution of the x and y coordinates determined by vision clearly depend on pixel size. More problematically, angular resolution depends on both pixel size and image width. Our scene geometry assumptions, however, imply that the relationship between image and pose resolution is constant.
- *reliability*: Depending on the quality of the primary position estimate discussed later, and the vision update rate, a stringent requirement on resolution can result from the need to track the mosaic. An angular error in the pose computed in a given cycle causes a position error in the location of the search window in the next cycle. This matter is analyzed carefully later in this section.

2.2.3 Storage

For a pixel size δ of 2.5 mm, Figure 3 shows that one-tenth of a square kilometer of area can be stored in 10 GByte of memory. This amount of storage corresponds to a single contemporary DVD ROM. Memory requirements drop quadratically if the pixel size can be increased.

In a realistic manufacturing setting, vehicles are limited to specific guidepaths perhaps a meter wide, so the graph also indicates that 100 Km of guidepath (10^5 square meters) can be stored in only 10 GByte of memory.

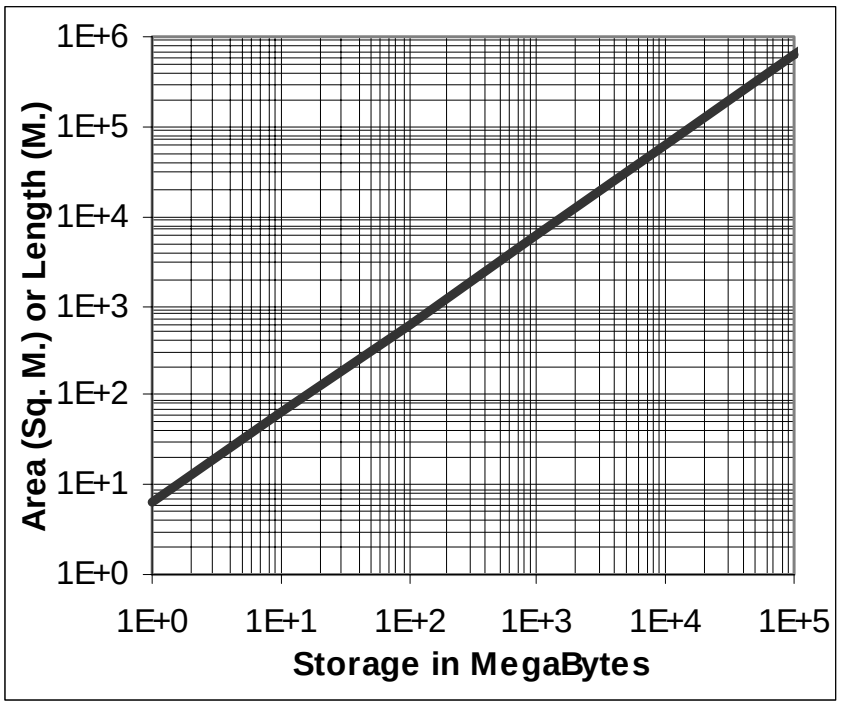

Figure 3: Storage Requirements. Up to 100 Km of guidepath can be stored at 2.5 mm resolution in 10 GByte of memory.

2.3 Texture

This section discusses the existence of appropriate texture in typical floor imagery and evaluates some representative imagery. Figure 4: shows images of floor scenes in manufacturing facilities which exhibit typical textured patterns.

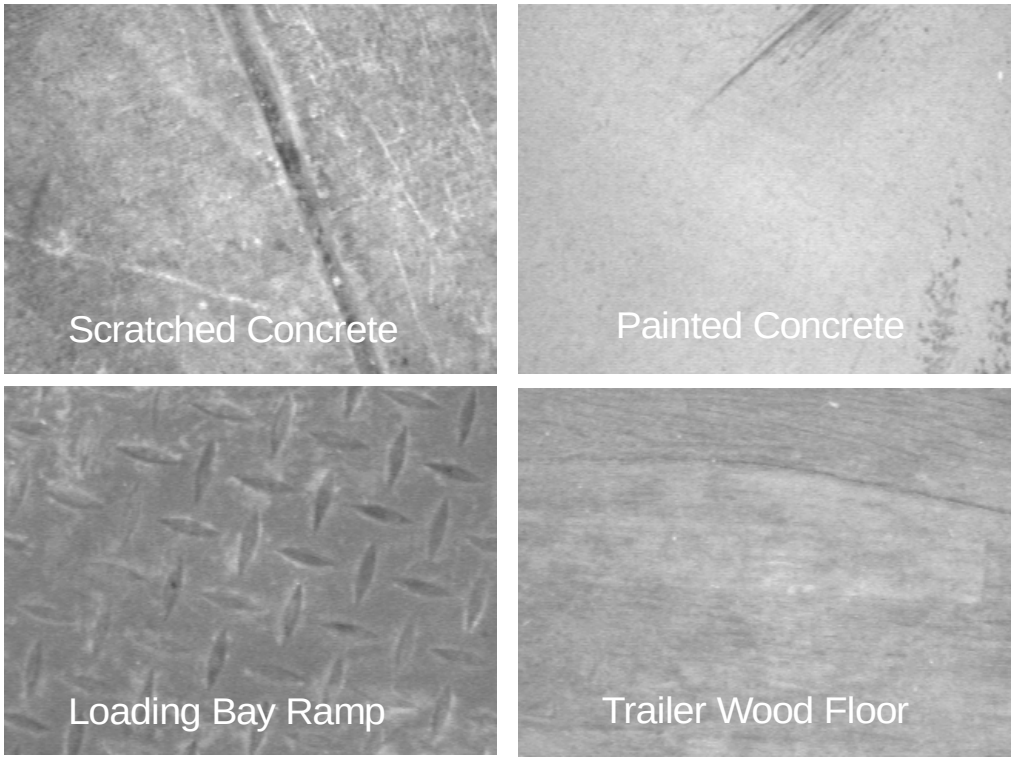

Figure 4: Manufacturing Plant Floor Imagery. Many man-made surfaces exhibit locally unique texture.

Much of the texture encountered is fairly persistent because it is due to mechanical wear and tear creating physical texture of the surface. In cases of repetitive texture, such as the ramp image, such processes can often introduce uniqueness.

2.3.1 Unambiguous Signal

A very direct test of whether or not an image, or part of an image, contains sufficient texture for template matching is to compute the auto correlation of candidate templates as a function of template displacement in all directions from its nominal position. A template is suitable for matching when:

- it possesses sufficient texture,
- it has a high cross-correlation surface peak,
- it has no competitive peaks in a neighborhood.

All conditions, when met together, will imply good noise rejection. Texture in the template implies that a local maximum correlation will exist. Many metrics of texture have been proposed, but intuitively, measures of local intensity variation, edginess, or gradient magnitude will suffice for a qualitative assessment.

Good cross-correlation implies a high degree of similarity between the template and the candidate mosaic region [5]. It is important to recognize that this second condition does not imply the first because even relatively textureless regions may correlate perfectly. Of course, any template *auto-correlation* surface peak always has a value of unity, so it is the normalized *cross-correlation*

between the image template and the mosaic that matters.

Uniqueness implies a low probability of false matches. A good measure of this third criterion is the difference in height between the highest and the second highest auto correlation peak in the search neighborhood. When the difference is large, it should be possible to correctly match the template even in the presence of substantial random noise because significant noise would be necessary to lower the higher peak and/or raise the lower peak.

2.3.2 Uniqueness in a Representative Concrete Floor Image

Figure 5: presents a quantitative assessment of the suitability of a concrete floor image using expensive, definitive, brute force techniques that would not be justified in a real-time implementation.

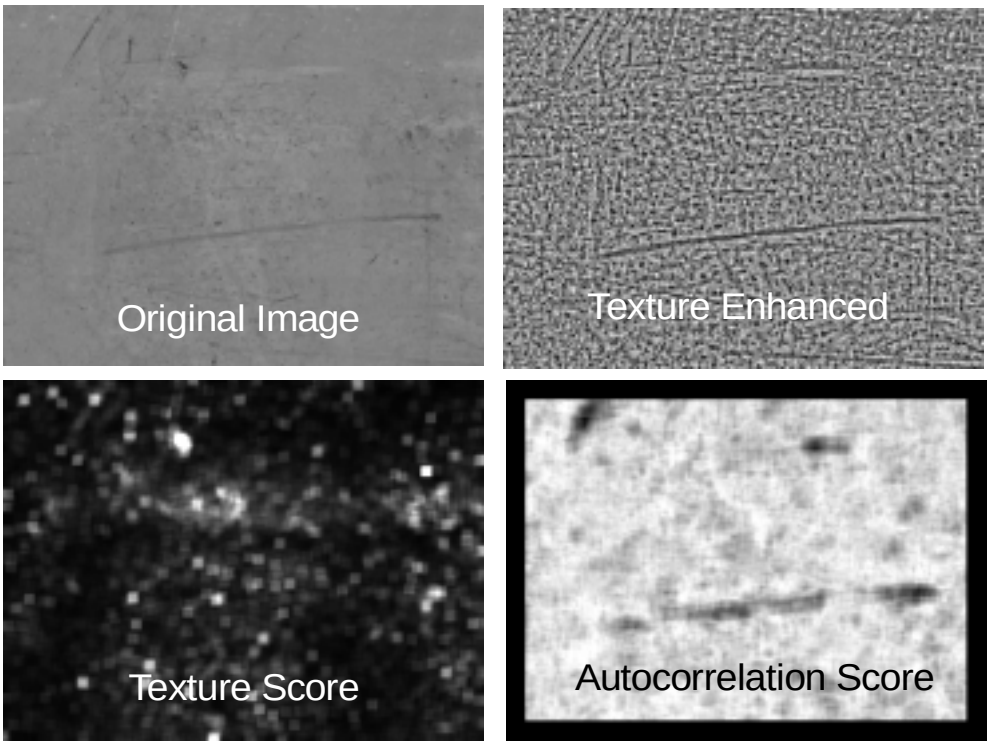

Figure 5: Analysis of Typical Floor Image. While it may subjectively seem that this concrete floor image lacks sufficient texture, it is actually acceptable for a 9X9 template size at every single pixel in the absence of noise.

Of course, one image tells us little about the general situation. However, one image can show that subjective evaluations of texture can be overly conservative. The goal is to show that the entire image is suitable for use as template features of some fixed size in some fixed local neighborhood.

The original image shows the nicks, cracks, and scratches that might be expected due to normal wear and tear. Point imperfections, which are particularly valuable for localization, are apparent as well. The texture enhanced image and texture score image are generated using the statistical normalization and gradient covariance algorithms described later in section . One can verify that

whiter pixels in the texture score image correspond to features of higher texture due to imperfections in the floor.

The auto correlation score image is generated from the difference of the highest and the second highest auto correlation peak in a 17 X 17 search window centered at each pixel. In general while tracking the mosaic, the search window size must always equal or exceed the accumulated error between vision fixes, estimated in section 2.5.4.

Let C_1 be the correlation score of the highest peak (which is unity for auto correlation). Let C_2 be the correlation score of the second highest peak (or it could be simply the highest score outside the radius of the highest peak). Then the auto correlation score image is a scaled version of:

$$\text{UniquenessScore}[i,j] = (C_1[i,j] - C_2[i,j])$$

For the test image, there are no pixels for which $(C_1 - C_2) < 0.2$ and fewer than 1% have $(C_1 - C_2) < 0.4$. Hence, this image is likely to be an excellent image for the purpose of template matching.

2.4 Persistent Appearance

Robustness to changes in appearance can be related to the stability of the uniqueness score introduced above under corruption of the original image. One type of analysis that can be done is to deliberately corrupt an image (simulating dirt, for instance) and then to cross-correlate the result with the original. This analysis is similar to that performed in [39].

The goal of this section is to show that feature matching in the previous relatively featureless image can be made robust to random noise when feature selection is augmented by a search for high texture regions. A preprocessing search for high texture pixels is a good idea for two reasons:

- Noise immunity is correlated with high texture.
- More thorough tests are too expensive to perform.

For example, computation of our texture score for a 160 X 120 image requires 70 milliseconds, whereas the correlation score requires a full 3 minutes.

Figure 6: illustrates the result of cross-correlating a pristine mosaic against a noisy image. A 17 X 17 search window is used, and a peak that is mislocated by 8 pixels is deemed a false positive. The threshold of 8 is chosen because the radius (sharpness) of the correlation peak is always related through common computations to the size of the correlation window in addition to inherent texture.

The curve labelled “texture threshold” depicts the highest texture score of any false positive at the indicated injected noise level (as determined from its deviation from ground truth). In this case, the vertical axis is proportional to the texture score. By definition, no region whose texture score exceeds the threshold generates a false positive. Such regions are safe for matching at the indicated noise level.

The curve labelled “percent above threshold” is the number of image pixels whose texture score remains above this threshold. In this case, the vertical axis represents a percentage.

Clearly as the threshold rises, the percentage of regions above threshold must lower. The fact that random noise of magnitude of 25% of the maximum intensity is required to render the highest texture threshold useless predicts a high level of noise immunity for high texture pixels.

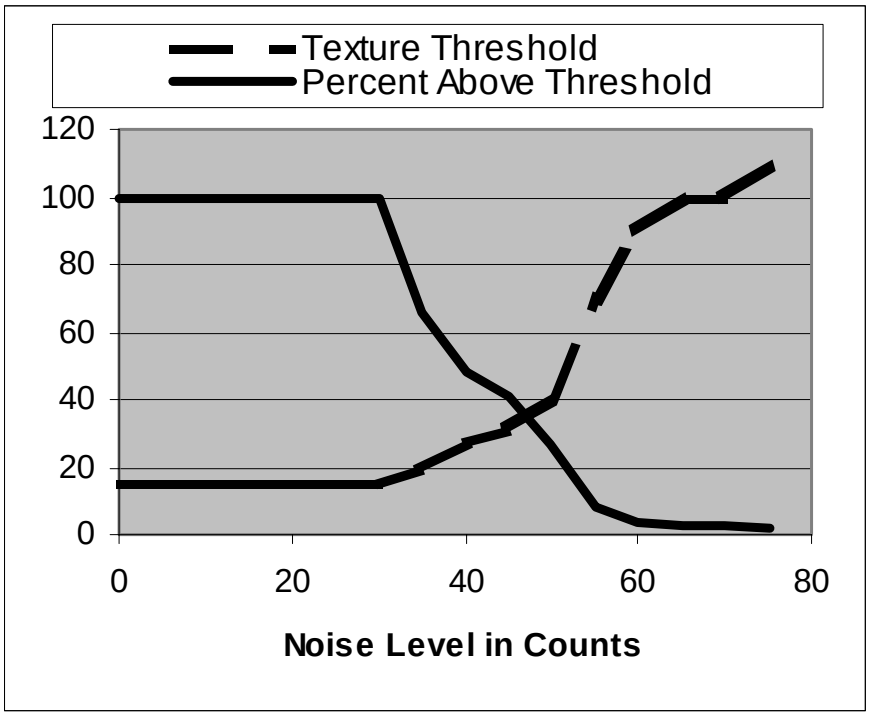

Figure 6: Noise Immunity. For a maximum grey level value of 255, it takes random noise with a standard deviation of 60 counts applied at every pixel in a template before the highest texture regions in the image generate false positives.

Of course, in addition to feature prequalification, another reason for this level of noise immunity in the face of little absolute texture is the surface-normal orientation of the camera optical axis. If necessary, the feature size can be increased to levels that impart high noise immunity without suffering from distortion problems.

This analysis concentrates on (Gaussian) random noise because it impacts the reliability of matching. Errors in motion estimation are another matter entirely - impacting the reliability of search. Such errors are analyzed next.

2.5 Processing Requirements

It remains to be shown that a vehicle moving at a given speed can support the processing necessary to update its position continuously using a mosaic-based positioning system.

2.5.1 Processing Required To Localize N Templates -Fixed Search Radius

Let us suppose that normalized cross-correlation is the matching mechanism used and that the time required to normalize the image can be neglected. For a template of size $w \times w$, it clearly takes at least w^2 operations to compute its cross-correlation coefficient with another template of equal size. If various overhead operations such as array indexing are included, let there be K operations

required per pixel in the template to arrive at Kw^2 operations per template correlation.

If an entire region of size $W \times W$ is to be searched, the careful reader can verify that it is not possible to reuse any computations, so the required processing to search for a match to a single template is KW^2w^2.

Processing two templates per image is a mathematical minimum to provide sufficient constraint on 2D camera pose. However, in order to determine the location robustly, it will likely be necessary to process more than this. Let's assume that N templates are necessary. Hence, computation of a position fix will require:

$$flops = NKW^2w^2$$

If a position fix is required every T_{cyc} seconds, then the required computational power devoted to template matching is:

$$f_{cpu} = \frac{flops}{T_{cyc}} = \frac{NKW^2w^2}{T_{cyc}}$$

This result can be rewritten in more convenient form. First, in terms of the number of correlation operations ("correlation ops") required per unit time:

$$f_{corr} = \frac{f_{cpu}}{K} = \frac{NW^2w^2}{T_{cyc}}$$

Second, in terms of the number of template matches required per unit time:

$$f_{templ} = \frac{f_{corr}}{w^2} = \frac{NW^2}{T_{cyc}}$$

Third, in terms of the number of units of search area tested per unit time:

$$f_{search} = \frac{f_{templ}}{N} = \frac{W^2}{T_{cyc}}$$

The variation of processing with search radius W and cycle time T_{cyc} has been left intact in order

to address below some mechanisms for determining them.

2.5.2 Intrinsic Processing Requirements

In order to not get lost, the area searched per unit time by vision must equal or exceed the error that has accumulated per unit time. In practice, both the search radius and the time period of vision cycles are somewhat arbitrary - provided the above condition is satisfied - but we will shortly show that an optimum value for both can be rationalized. Before doing so, it is necessary to determine models for the accumulation of error with time and distance.

2.5.3 Odometry Error Accumulation Model

Consider the problem of predicting the time evolution of tracking error after some particular fix. Here, the tracking error is the difference between the predicted position of a template and its actual position in the mosaic. The predicted position depends on the results of the last fix, and the odometry error since then. The actual position depends on the true position in the scene and the error in the mosaic. Hence, the total difference is the sum of many effects.

Let us assume that a dead reckoning system is used to compute an estimate of where the camera is, and that the estimate is used to compute the center of template search regions. Further, assume that the dead reckoning system computes heading by integrating differential angular changes evident from the differences in the encoder readings of two wheels.

Let a random noise component whose variance is proportional to distance be present on each of two wheel encoders. This quantity represents a lumped parameter for all sources of random error and takes the form:

$$\sigma_{ss} = \gamma \cdot ds$$

Let a scale error proportional to distance be used as a lumped parameter to represent the accumulated effect of all forms of systematic encoder errors such as changes in wheel radius, calibration errors, wheel slip etc. This quantity takes the similar form:

$$\Delta s = \alpha \cdot ds$$

Although the errors take similar forms, they propagate differently since systematic errors add directly whereas random ones add in the root mean square sense.

Perturbation analysis (detailed in [24]) can be used to show that, in the case of integrated heading, for straight reference trajectories, the greatest error accumulates in the direction transverse to the direction of travel for both random and systematic sources.

Let S denote the total distance travelled, L denote the wheel separation, γ denote the random error gradient, and α denote the systematic error gradient. The accumulated odometry errors take the

form:

Table 1: Accumulation of Odometry Error - Integrated Heading and Straight Reference Trajectory

Symbol	Meaning	Value
σ_{xx}	Accumulated random transverse error	$\frac{2\gamma S^3}{3L^2}$
$\sigma_{\theta\theta}$	Accumulated random heading error	$\frac{2\gamma S}{L^2}$
Δx	Accumulated systematic transverse error	$\frac{\alpha S^2}{2L}$
$\Delta\theta$	Accumulated systematic heading error	$\frac{\alpha S}{L}$

2.5.4 Growth of Error Between Fixes

The results of Table 1 can be used to predict the growth of error between fixes. The error in predicted position of the camera depends on both the error in the start pose due to the last fix, and the error incurred by odometry since the last fix. Additionally, any separation of the camera from the reference point of odometry further amplifies the effect of the angular components of error from both sources (as does the deviation of templates from the center of the image).

Suppose the vehicle starts at pose P_0 and it is perturbed by errors in the first position fix of Δx_0 and $\Delta\theta_0$ to pose P_0'. Thereafter, odometry is used to determine the motion that occurs over an excursion of length S. In the absence of any error, the vehicle pose would be computed to be P_f and the corresponding camera pose would be C_f. These are the ground truth terminal poses. However, we will model the impact of systematic and random error sources which put the computed vehicle and camera terminal poses at P_f' and C_f'.

Errors will be assumed to be small enough to propagate linearly. The total heading error is the sum of the initial and the odometry error.

$$\Delta\theta = \Delta\theta_0 + \Delta\theta_{odom}$$

The total crosstrack error is the sum of the initial crosstrack error, the effect of the initial heading error acting over the distance travelled, the crosstrack error due to odometry, and the effect of the total heading error acting over the distance D from the vehicle frame to the camera frame.

$$\Delta x = \Delta x_0 + S\Delta\theta_0 + \Delta x_{odom} + D\Delta\theta$$

We will estimate the random components of error as 3 times the standard deviation. Using expressions from Table 1, we therefore have:

Substitution for the crosstrack error leads to its accumulation with distance:

$$\Delta\theta_{odom} = \frac{\alpha S}{L} + 3\sqrt{\frac{2\gamma S}{L^2}}$$

$$\Delta x_{odom} = \frac{\alpha S^2}{2L} + 3\sqrt{\frac{2\gamma S^3}{3L^2}}$$

$$\Delta x(S) = \Delta x_0 + S\Delta\theta_0 + \frac{\alpha S^2}{2L} + 3\sqrt{\frac{2\gamma S^3}{3L^2}} + D\Delta\theta_0 + D\frac{\alpha S}{L} + 3D\sqrt{\frac{2\gamma S}{L^2}}$$

This expression contains seven terms respectively corresponding to:

- initial crosstrack error,
- initial heading error acting over distance travelled,
- systematic odometry crosstrack error,
- random odometry crosstrack error,
- initial heading error acting over camera offset,
- systematic odometry heading error over camera offset,
- random odometry heading error over camera offset.

The situation is summarized in Figure 7.

This result can be quantified by assuming values for the parameters. The following table summarizes the assumptions used.

Table 2: Error Accumulation Model Parameters

Symbol	Meaning	Value
δ	Pixel size	2.5 mm
Λ	Image size	100 δ

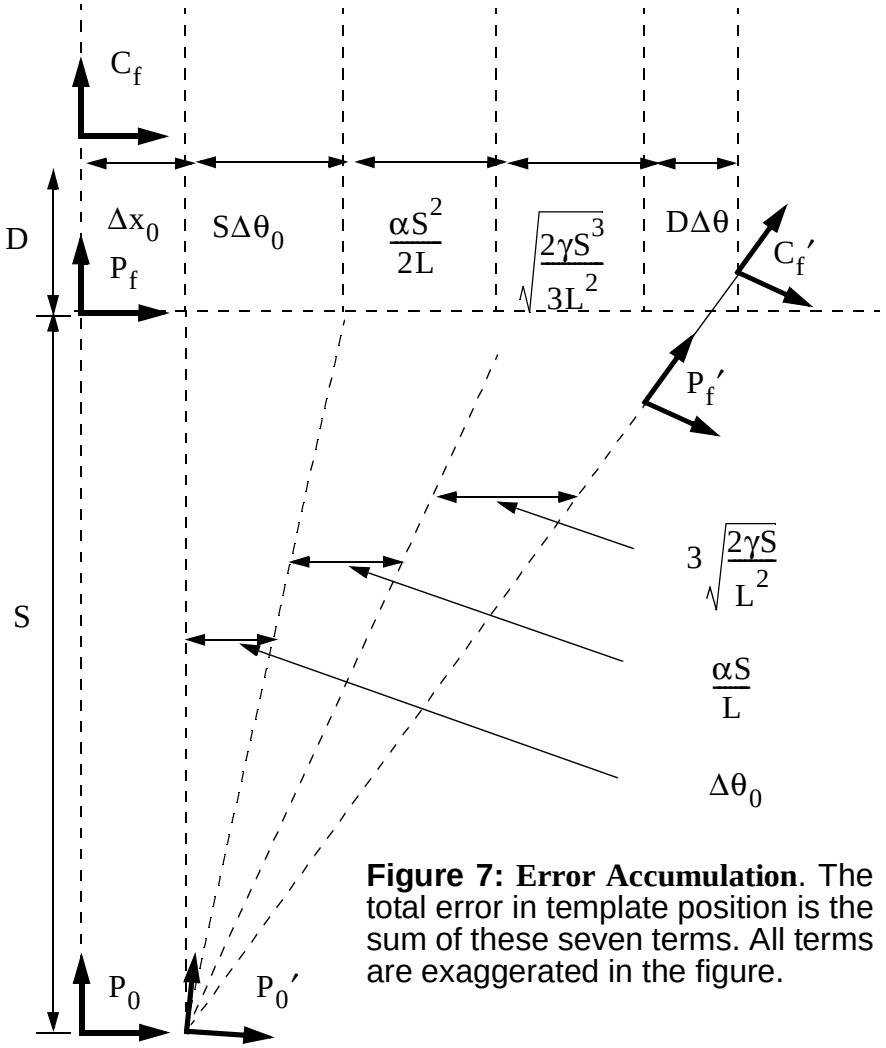

Figure 7: Error Accumulation. The total error in template position is the sum of these seven terms. All terms are exaggerated in the figure.

Table 2: Error Accumulation Model Parameters

Symbol	Meaning	Value
α	Systematic Encoder Error Gradient	0.01
γ	Random Encoder Error Gradient	0.0001
L	Wheel tread	1.0 m
D	Camera Offset	0.5 m
$\Delta\theta_0$	Initial angle error	δ/Λ
Δx_0	Initial crosstrack error	δ

These assumptions give rise to the error accumulation with distance indicated in Figure 8.

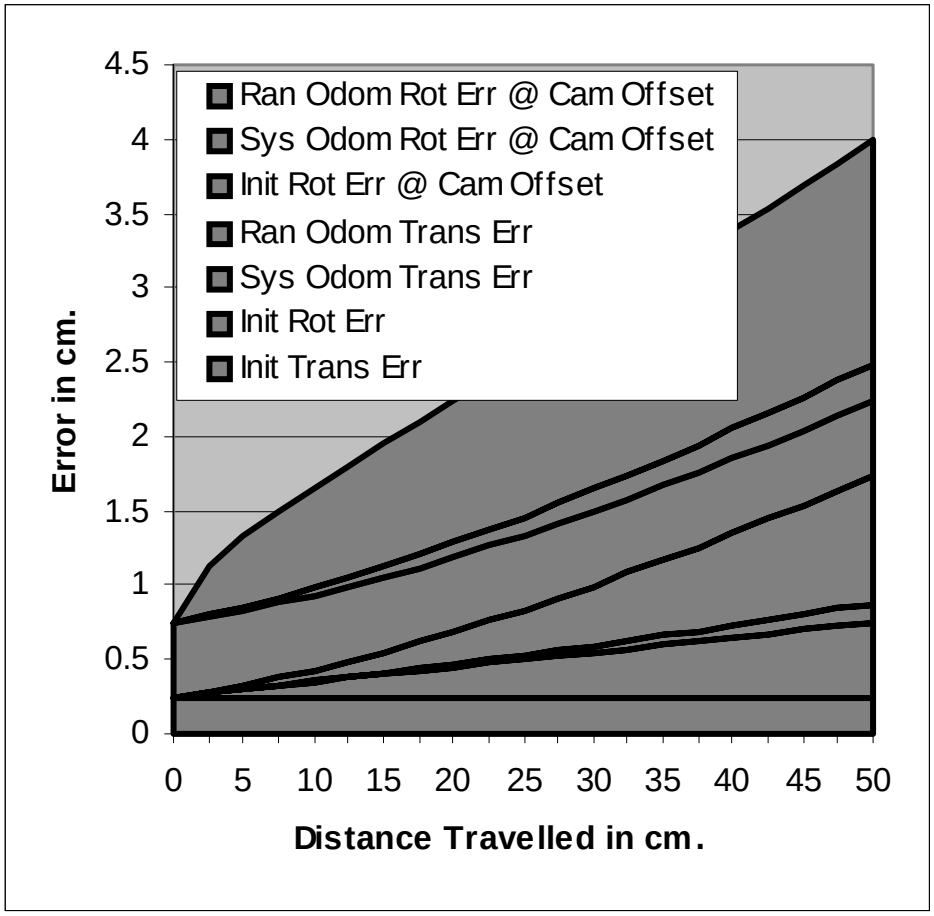

Figure 8: Error Accumulation. Various systematic and random error sources combine to produce the total error in estimated position of a template as shown. The legend is listed in the order that the regions are stacked.

Such an error model is far more realistic than a simple percentage of distance travelled. Note, for example, that at a distance of 10 cm, while the random error gradient α is only 1% of distance, the total error is 15% of distance. This discrepancy is due mostly to the assumed initial heading and crosstrack error.

2.5.5 Processing Required for Tracking - Variable Search Radius

Consider designing a mosaic-based localization system for which the search "radius" $W/2$ is adjusted dynamically to equal the estimate of error $\Delta x(S)$ at any given time. If the vehicle travels at constant velocity V, and the pixel size is δ, then the distance travelled is easily related to the cycle time and the search diameter in pixels becomes:

Our earlier expression for the processing required to maintain visual lock becomes:

The cycle time is still a free variable since we can choose to cycle at any rate regardless of vehicle

$$W = \frac{2\Delta x(S)}{\delta} = \frac{2\Delta x(VT_{cyc})}{\delta}$$

$$f_{search} = \frac{W^2}{T_{cyc}} = \frac{4\Delta x^2(VT_{cyc})}{\delta^2 T_{cyc}}$$

speed. If we reuse our assumptions for error accumulation parameters, then a curve results for every value of vehicle speed for a given fixed cycle time. The growth of processing requirements with speed for four specific cycle times is shown in Figure 9. In a manner explained next, faster or slower cycles may be better depending on the speed.

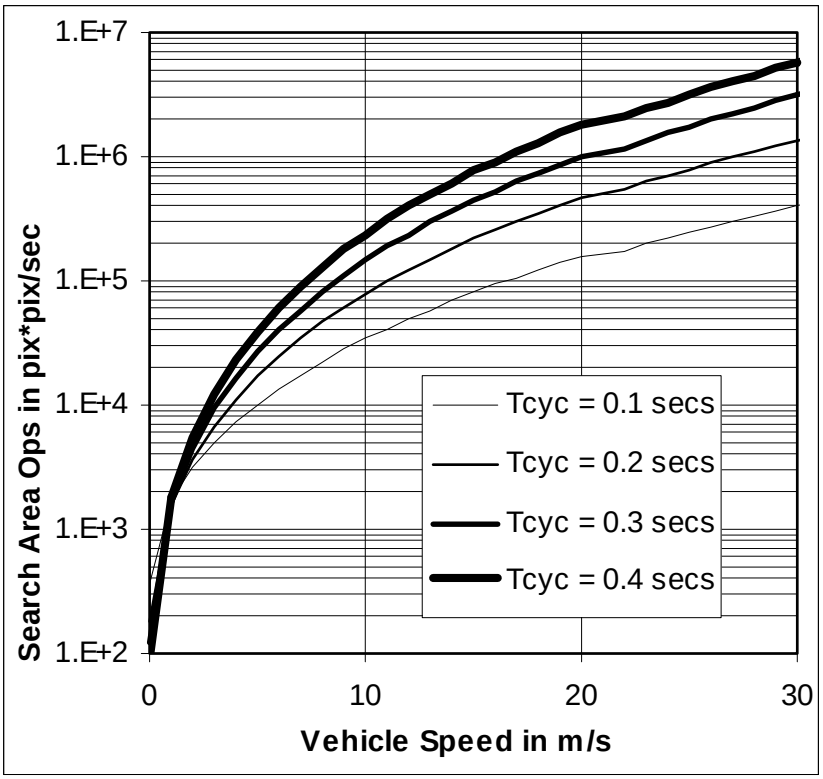

Figure 9: Processing Requirements. Increase with vehicle speed. From bottom to top on the right, the 4 curves are for T_{cyc} of 0.1, 0.2, 0.3, and 0.4 secs.

2.5.6 Update Rate For Minimum Processing Requirements

The dependence of processing requirements on speed is nonlinear, and hence it potentially has an extremum. As cycle time is increased, the required processing rate nominally goes down because

the number of searches conducted per second is smaller. However, we have tied the size of the search radius to speed in such a way that it simultaneously drives processing requirements in the opposite direction.

Due to build up of error over time, the size of those searches increases. Hence, the relationship is a rational polynomial in T_{cyc} as was noted in the former expression:

$$f_{search} = \frac{W^2}{T_{cyc}} = \frac{4\Delta x^2 (VT_{cyc})}{\delta^2 T_{cyc}}$$

Although this expression is a function only of T_{cyc}, it is tedious to differentiate it to find the extremum. Numerically, however, it is easy to find the best cycle time for a given speed. Figure 10 gives processing requirements as a function of cycle time at a fixed speed of 2.5 m/s.

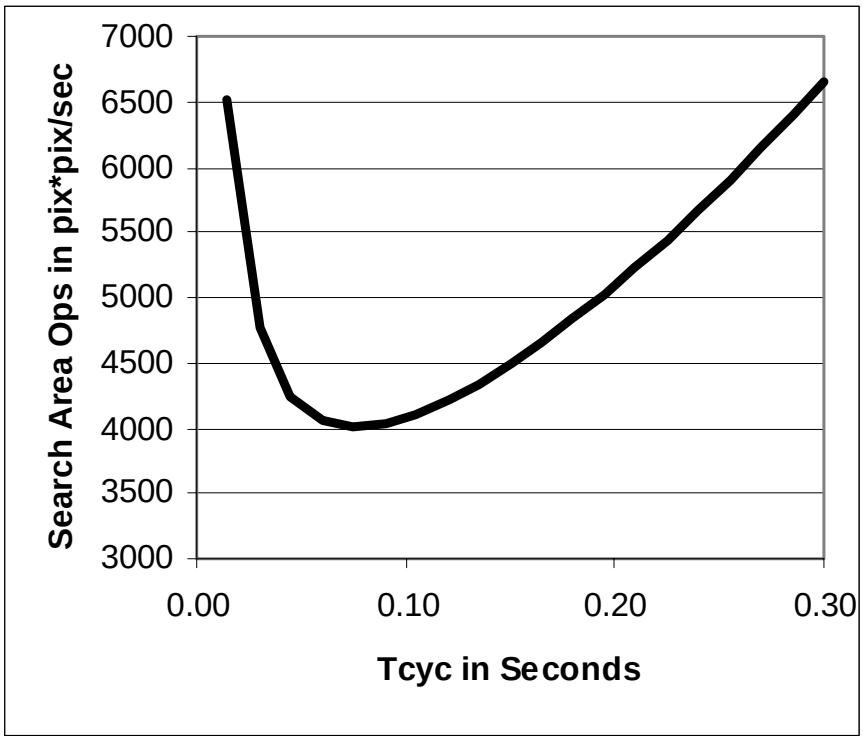

Figure 10: Minimum Processing Requirements. Because processing requirements are nonlinear in cycle time, an optimum cycle time exists. This curve is for a speed of 2.5 m/s.

This analysis of tracking requirements is similar to [46] but here, a noise floor has been added. Our noise floor models the fact that the last fix has some error in it. This component of error is not reduced by faster tracking updates so it penalizes faster updates and moves the optimum update rate away from the camera frame rate toward a finite cycle time. In a more general setting, feature distortion would place limits on how much cycle time can be increased before feature matching becomes problematic.

2.5.7 Dependence of Processing On Odometry

Given the introduced dependence of search radius on error, it is instructive to evaluate the

quantitative impact of odometry. Figure 11 shows the results of varying the systematic odometry error gradient α from zero to unity while operating at the speed and associated optimal cycle time of Figure 10. The random error gradient γ is zeroed for the whole graph. Hence, zero for α represents perfect odometry whereas unity represents lack of odometry where features are naively predicted to be in the last location reported by vision. For our error model, and an assumed speed of 2.5 m/s, odometry has the power to reduce processing requirements by 2 orders of magnitude.

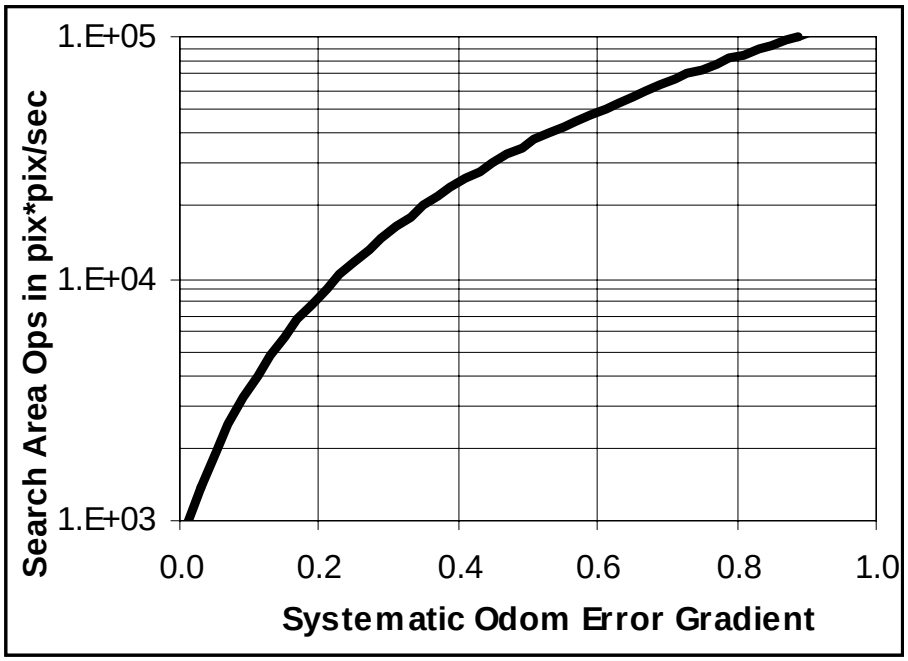

Figure 11: Dependence of Processing on Odometry. Required processing is reduced up to 1.5 orders of magnitude with odometry at a speed of 2.5 m/s.

3. Performance and Reliability Analyses

Certain aspects of performance and reliability are structural. That is, they are properties of the approach rather than of the environment, the hardware, the algorithm or its implementation. This section explores some figures of merit that are particularly relevant to our approach.

In the process, it becomes clear that our scene geometry assumptions, a prior model, and odometry aiding lead to levels of tracking performance not possible in visual odometry and visual tracking. Several different regimes of operation exist which indicate clearly why mosaic-based localization achieves relatively high levels of tracking performance.

3.1 Projective Mapping

In mapping quantities in the scene to their associated quantities in the image plane, scene geometry dependence and projective loss of information can, of course, complicate matters. Figure 12 indicates the simplest case of motion confined to a single axis parallel to the image plane and defines notation for this section.

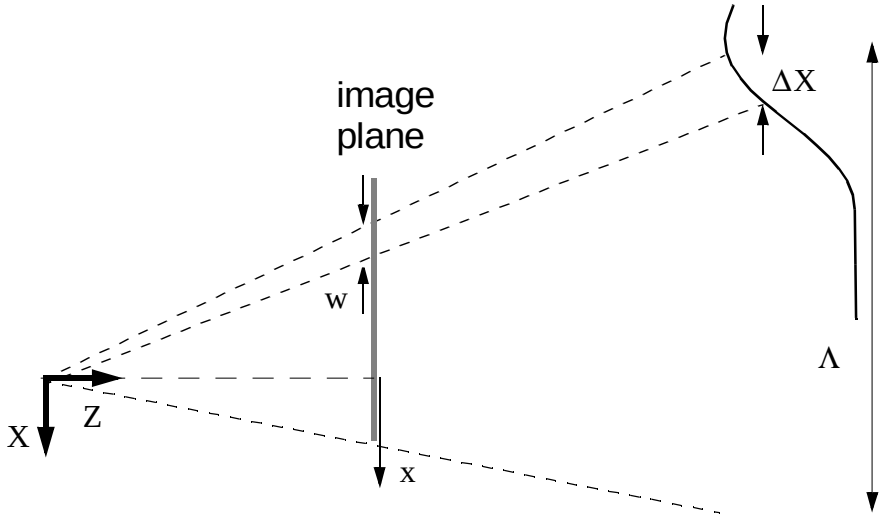

Figure 12: Simplified Projection. Notation for a projection from 2D to 1D.

Let x denote the image coordinate and X denote the corresponding scene coordinate while Z denotes depth. If the focal length is f, the basic projective mapping to the image plane is elementary:

$$x = (f/Z)X$$

In this restricted case, if depth is constant over a small motion in the scene, differential motions and hence velocities scale linearly from the scene to the image:

$$\Delta x = (f/Z)\Delta X$$

and depth in units of focal length is an important scaling parameter. Since motion parallel to the image plane is the worst case, limits on the feature velocities that can be tracked give rise to corresponding limits on associated camera-to-scene relative velocity.

3.2 Performance Attributes Related to Geometry

Of course, one of the many implications of the above mapping is that points at different depths have different image velocities, and therefore, depth gradient in a template implies distortion as the template moves. If we also allow depth to vary across a template, the total differential is:

$$\Delta x = (f/Z)\Delta X - (f/Z^2)Z_X \Delta X$$

and a dependence on depth gradient in the scene Z_X (infinite at occluding boundaries) is introduced. Dividing by a small time increment it is clear that while the first term indicates template motion due to camera motion, the second indicates a distortion effect.

3.2.1 Distortion Speed Limit

When the range to features varies substantially over an image, severe limits on speed of tracking can be introduced by the distortion and/or occlusion of features from frame to frame. A small patch of scene of width ΔX projects onto an image template of width:

$$w = (f/Z)\Delta X$$

As the template moves across the image due to camera motion at speed V, for a time period of T_{cyc}, its width changes by:

$$\Delta w = (f/Z^2)Z_X V T_{cyc}$$

Hence, the change in template size in relation to size is:

$$\frac{\Delta w}{w} = \left(\frac{Z_X}{Z}\right)\frac{VT_{cyc}}{\Delta X} = \left(\frac{Z_X}{Z}\right)\left(\frac{u}{w}\right)T_{cyc}$$

where u is the template velocity in the image plane. If distortion is to be reduced in order to make feature matching easier, the only way to do so for given geometry is to reduce the nondimensional $u/(w/T_{cyc})$ which represents the template velocity in units of templates per cycle. If template size is determined by texture content or available computation, this can only be accomplished by reducing speed, or cycle time.

Of course, the expression also shows that distortion is eliminated if there is no depth gradient across the template. When geometry is such that features can be tracked across a significant portion of the image in a single cycle, another limit comes into play. This elementary observation is important here because our scene geometry assumptions allow us to break the distortion speed barrier.

3.2.2 Overlap Speed Limit

In many visual tracking applications, tracking features in successive frames implies a fundamental speed limit induced by the geometric constraint of overlapping fields of view. If the image width at the feature depth is Λ, and β is the fraction of image overlap required for matching, the camera speed V must satisfy:

The velocity of the camera in units of images moved per cycle is thereby limited to a value somewhat less than unity. If no feature is to be skipped over, the calculation must be performed for the feature at the minimum depth - whose velocity in the image is highest.

$$V < \left(\frac{\Lambda}{T_{cyc}}\right)(1 - \beta)$$

This observation is important because the use of prior models such as mosaics allow us to also break this speed barrier when tracking.

3.2.3 Geometric Instability

Exceeding the overlap speed limit has important implications on the stability of tracking. Suppose for simplicity that depth is constant, and that the camera moves parallel to the image plane. The heading can be determined from the positions of two features sufficiently separated in the image. Let one feature be mislocated by an error ΔX_k. The resulting effect on the computed heading is:

$$\Delta\theta_k = \Delta X_k / \Lambda$$

Suppose further that the camera travels a distance S before another visual fix is attempted. At this point, the error of the original fix causes a position error normal to the direction of travel at the new position of:

$$\Delta X_{k+1} = (S/\Lambda)\Delta X_k$$

Since the nondimensional ratio of distance travelled to image width exceeds unity beyond the overlap speed limit, visual tracking becomes *unstable*. Given that there are many other sources of error that affect the predicted positions of features in the next image, instability is actually reached earlier than the above relation implies.

An important implication of this result is that errors of acceptable magnitude in a given visual fix can cause loss of visual lock in the next cycle where it would have been possible to recover at lower speeds. Such errors might be due, for example, to false feature matches or distortions of the mosaic. No amount of computational speed can help, and searching a radius larger than necessary can mean asking for trouble.

Specifically, even if a search diameter of W_{max} can be sustained computationally per cycle, it can be prudent to limit the impact of feature matches beyond a safety diameter such as

$$W' = (\Lambda/S)W_{max}$$

because if the match is wrong, loss of lock is guaranteed in the next cycle. If, however, the match is correct, not using it will also cause problems - but this scenario indicates speeds too close to computational limits.

Pragmatic strategies to manage this issue include redundant sensing of intervening motion,

estimates of accrued error, higher update rates, consistent mosaics, robust matching, and outlier rejection in pose determination.

3.2.4 Template Rotation Search Threshold

While templates do not distort in the absence of depth gradient, they do rotate. Since they rotate with the camera, their translation and rotation in the image plane must, however, be consist with rigid body motion. If a template search radius of $W/2$ in the image plane is being searched, the system fundamentally makes the assumption that template translational error is less than this. For templates separated by the image width, the associated maximum translation at the edge of the template due to its own rotation is:

$$\Delta\theta = (Ww)/(2\Lambda)$$

This nondimensional is important because it allows us to ignore template orientation altogether because it cannot deviate enough from expectations to matter. When it is less than unity, template search can be constrained to translation only and order of the search process is reduced from a 3D one to a 2D one.

3.2.5 Impact on Mosaic Smoothness Requirements

It is clear that a mosaic must be free from registration errors and false intensity gradients everywhere in order to support reliable template matching. However, the preceding results imply a need for geometric consistency as well. First, if the assumption of 2D rigid body motion of templates is to remain valid, areas on the order of size of an image must remain relatively undistorted - or affine transforms may be required. Further, if the heading derived from one fix is used to assist odometry in estimating the location of the next fix, then areas on the order of size of the distance between images must remain relatively undistorted so that, for example, a line between three features in the scene remains undistorted in the mosaic.

3.3 Performance Limits Related to Processing

Another class of performance limits are related to the speed of the processor. Recall that earlier, we expressed processing required as a function of the search radius $W/2$ and cycle time as:

$$f_{search} = \frac{W^2}{T_{cyc}}$$

3.3.1 Processor Limited Speed

Since error depends on the motion between images, and required processing depends on error, processing required depends on motion. The results of Figure 9 summarize a detailed analysis, but a simpler closed-form result is sought here for the inverse relationship of speed on processing.

Let the error that accumulates between images be given solely, and naively, by a fraction of

distance:

$$\Delta X(S) = \alpha S$$

Setting the search radius $W/2$ to this and rearranging terms:

$$f_{search} = (2\alpha S)^2 / T_{cyc} = 4\alpha^2 V^2 T_{cyc}$$

Indicating a quadratic dependence of processing on speed. Note that using no estimate of motion between images implies $\alpha = 1$. If a dynamic model can generate a better estimate equivalent to $\alpha = 0.1$, required processing drops two orders of magnitude. If odometry can be used to generate an estimate equivalent to $\alpha = 0.01$, required processing is reduced two more orders of magnitude. This amounts to an intuitive explanation of Figure 11.

Solving now for the speed at which accumulated error equals processing available:

$$V = \sqrt{(f_{search} / T_{cyc})} / (2\alpha)$$

This result indicates that highest tracking speeds are achieved for the smallest cycle times and possible speed grows with the square root of available processing.

Further, in this simple model, sustainable speed is inversely proportional to error gradient α. As a result, the use of odometry might be expected to permit speeds up to 100 times the speeds achievable by searching from the last fix. However, more realistic models can be expected to reduce the predicted achievable speed significantly.

3.3.2 Safe Distance After Loss of Visual Lock

Fixed computing speed implies that a certain maximum feature search radius can be sustained per cycle.

$$\dot{\varepsilon} = \frac{W}{2T_{cyc}} = \frac{f_{search}}{2W}$$

where f_{search}, the number of feature comparisons computable per second is characteristic of the processor. A feature outside this radius will fail to be found and tracking will, at this point, fail. As was noted in [47]:

- This quantity has an optimum at some cycle time (the optimum in our case of a noise floor was computed earlier).
- Given a dynamic model with some number of derivatives (state variables), the radius can be related to the time series truncation error associated with the first neglected derivative.

While it is common to assume that T_{cyc} can be set arbitrarily within the limits of processing, this is not really the case. No features can be tracked when the camera is obliged to leave the mapped area or when occlusion, replacement, or obliteration of all texture in a relatively large region has occurred. This condition may occur in a particular vision cycle or a contiguous sequence of cycles.

For any given maximum sustainable search radius, the largest sustainable excursion without a position fix can be determined from graphs like Figure 8: by determining the distance at which the search radius is exceeded. For a rough rule of thumb, we can readily derive from the assumptions of this section that sustainable excursion grows with the square root of processor speed and that it

is, again, massively improved by odometry:

$$S = (\sqrt{f_{search} T_{cyc}})/(4\alpha^2)$$

Of course, if no processing in excess of the minimum required is available, a single missed image implies tracker failure.

4. Design

This section describes the major design decisions, overall architecture, and important algorithms. Our implementation is described for the specific purpose of AGV localization. Clearly, the implementation might differ significantly for other applications.

4.1 Position Estimator

The overall architecture of the position estimation system, of which mosaic-based localization is only a component, is as shown in Figure 13:

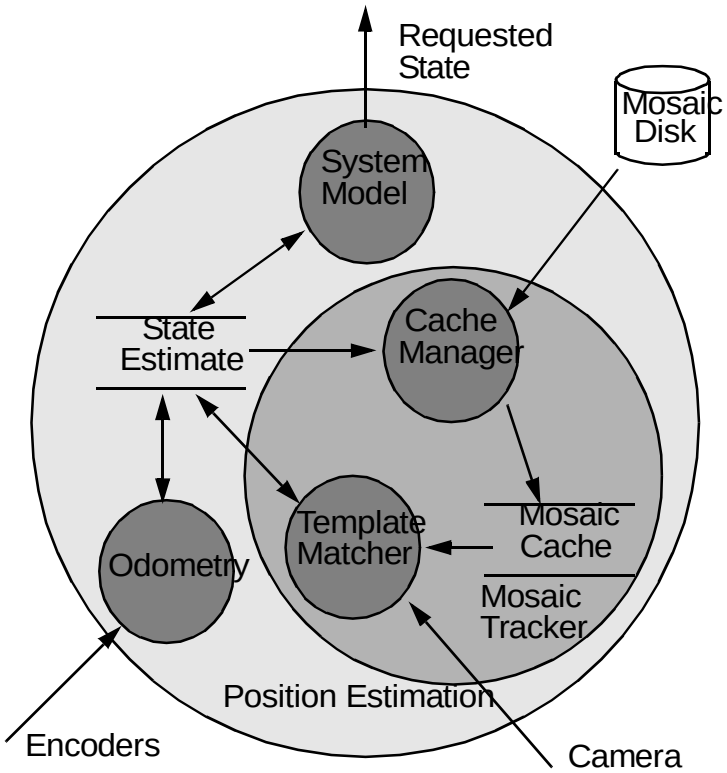

Figure 13: Localization System. The system model integrates the equations of motion to interpolate to the instant of time requested. The odometry system and mosaic tracking system provide two complementary estimates of state. For large scale mosaics, a cache is needed to store part of the mosaic in RAM.

As long as the system is operating, camera imagery is acquired as fast as processing can manage. Simultaneously, an odometry thread of execution continues to read the wheel encoders to provide position estimates between image acquisitions.

Position estimates between encoder readings are supplied by integrating the equations of motion under a constant linear and angular velocity assumption. This system model runs continuously whether or not state requests have been received.

The inner circle delineates the mosaic tracker. The job of the mosaic tracker algorithm is the core problem of mosaic-based localization. It must determine where the camera is over the map. Following the localization literature, this section will use the more generic term *map* to mean the mosaic.

4.2 Mapping and Localization Modalities

The position estimator is coordinated with a mapping process which operates in a number of useful modes.

4.2.1 Visual Odometry

When the camera is continuously moving over an unknown area, provided successive images overlap, visual odometry, perhaps aided by the encoders, can be performed. At such times, evolving position error is unbounded.

4.2.2 Automatic Mapping

While traversing unknown areas, provided successive images overlap, the acquired imagery can optionally be added to the mosaic. If it is, the position reported for the location becomes repeatable. While this process can produce mosaics automatically, they are not guaranteed to be globally consistent unless they are (both apparently and actually) acyclic.

4.2.3 Simultaneous Localization and Mapping

When both visual odometry and automatic mapping are being performed, the system is performing a restricted form of simultaneous localization and mapping [28]. Rendering mosaics globally consistent is an off-line process, so it is discussed in a separate section.

4.2.4 Automatic Map Updates

In principle, an image which appears sufficiently different from expectations, but is nonetheless confidently positioned, can be used to overwrite the data in the map to reflect a change in appearance.

4.2.5 Automatic Mode Switching

It is possible for the system to automatically switch from one mode to another. Conceptual logic is as follows:

```
if (the current image has minimal
   overlap with the mosaic)
   {
   add it to the mosaic;
   }
else if (the current image looks
   very different from the mosaic,
   but is confidently positioned)
   {
   overwrite the mosaic with the
   new image;
   }
else
   {
   compute pose and discard the image;
   }
```

4.3 Mosaic Tracker

The mosaic tracker is packaged as a stand-alone unit to permit it to be interfaced to a client system where it would serve as a terrain-aiding sensor package with embedded computing. It accepts an input state estimate and produces an updated one based on tracking the mosaic. Each execution of the mosaic tracker algorithm follows several steps. The following sections discuss each step in the process in the order they occur.

4.3.1 Input Transforms

The input state and uncertainty estimates must be transformed to account for three effects:

- *Camera Offset Compensation:* Incoming state and covariance must be adjusted to reflect the kinematic relationships between the camera and odometry.
- *Delay Compensation:* Poses are adjusted to compensate for delays of any sort between image acquisition and the state estimate.
- *Tracking Error Compensation:* Incoming poses are processed differentially to compensate for any long term drift between vision and incoming external estimate.

The last compensation arises because the need to track the mosaic is an entirely separate matter from the need to generate an optimal estimate of state. The mosaic derived estimate is not optimal, in general. Note, however, that failing to incorporate mosaic corrections may render the external estimate nonrepeatable.

4.3.2 Image Preprocessing

Incoming images are preprocessed to compensate for lens distortion and lighting variation. For the latter, statistical normalization is used. Each pixel is replaced by its suitably scaled, normalized deviation from the mean of its local neighborhood thus:

$$I' = (I - \mu_I) / \sigma_I$$

Figure 14 presents a concrete floor image and its texture enhanced derivative image.

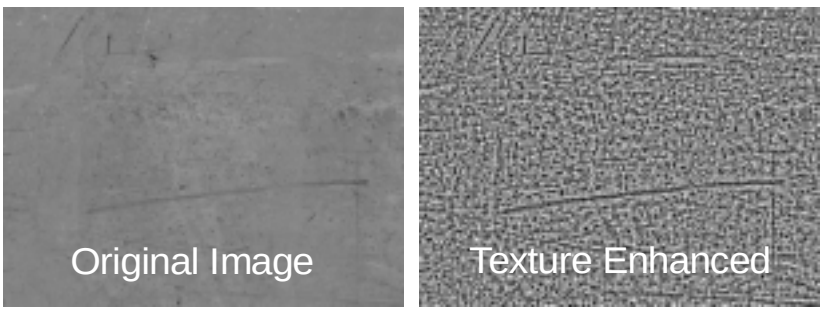

Figure 14: Normalization. This process removes most of the effects of image scale and bias differences between different images of the same floor area.

4.3.3 Texture Scoring

Let ∇x denote the intensity gradient in the x direction at a point in the image. For each pixel to be scored, for a neighborhood surrounding it, the smallest eigenvalue of the gradient covariance matrix [31] is evaluated:

$$\text{TexScoreDyadic} = \begin{bmatrix} \sum \nabla x \nabla x & \sum \nabla x \nabla y \\ \sum \nabla x \nabla y & \sum \nabla y \nabla y \end{bmatrix}$$

thereby rating potential features by their score in the direction of least texture. Better ways to find good features have been reported [39].

In general, for any given texture level, the smallest template that is still large enough to generate a unique surface peak is computationally most efficient. Although methods to dynamically adjust window size are available, we use a fixed window. Figure 15 presents a concrete floor image and its texture score derivative image.

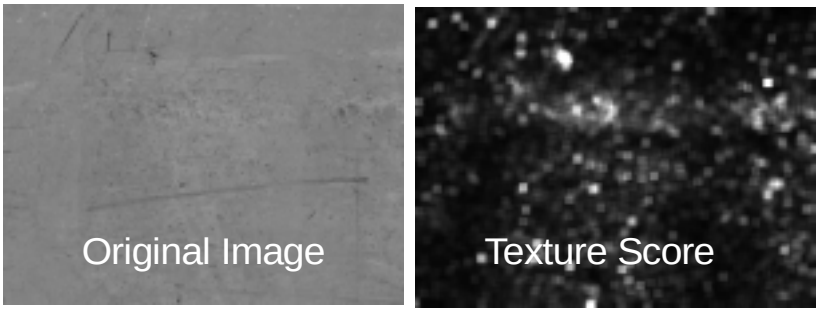

Figure 15: Texture Scoring. The texture scoring process identifies parts of the image possessing the most texture.

4.3.4 Feature Selection

For rectangular images, their overlap region is a convex polygon with at most 8 sides. In this region, features are selected based on their position and texture scores in the input image. This stage tries to simultaneously maximize the quality of the selected features and the distance between them (in order to enhance angular resolution). Once a feature is selected, its position in the mosaic is predicted, and a search region around its predicted position is extracted from the mosaic. Regardless of the data structure used for the mosaic, a coherent block of pixels surrounding each feature is most useful when template cross correlation is computed later in the cycle.

4.3.5 Template Matching

While many popular methods of computing similarity exist, we currently use normalized correlation. Let a template extracted from the image be of size $w \times w$ pixels. The mosaic will be searched for it inside a rectangular region of size $W \times W$. The two window sizes are independent. While w is determined by the amount of texture, W is determined by the current position estimate uncertainty. In our case, the template rotational search threshold computed in section 3.2.4 is not exceeded so it is not necessary to search for template orientation.

The *displacement* of each template is the difference between predicted and actual position in the mosaic. Displacement is determined by search for the peak in the cross correlation function between the template and the search region in the mosaic. If $T(x, y)$ is the normalized template intensity function and $M(x, y)$ is the normalized mosaic intensity function, then the normalized correlation, computed as a function of displacement (d_x, d_y), comes from the double integral:

$$C(d_x, d_y) = \frac{1}{w^2} \int_{-\frac{w}{2}}^{\frac{w}{2}} \int_{-\frac{w}{2}}^{\frac{w}{2}} T(u, v) M(u + d_x, v + d_y) du dv$$

Figure 16: presents a template and a search region from a concrete floor image, and the correlation between them as a function of two-dimensional displacement.

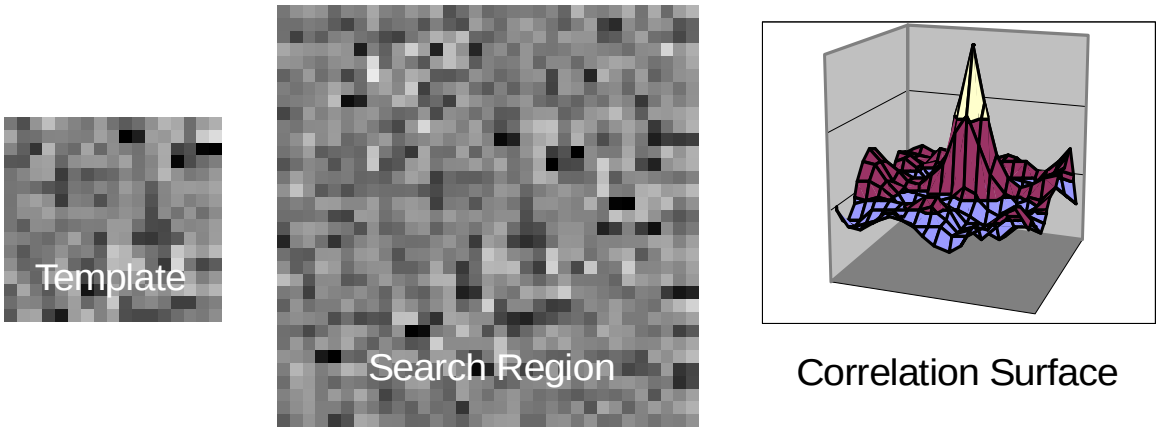

Figure 16: Principle of Operation. The translational pose error can be found as the position of the peak in a surface: the correlation surface as a function of displacement.

Visual examination will show that the template is present in the center of the search region. The output of the entire matching process is a point correspondence. For each selected template comprising a feature, its predicted position and observed position in the mosaic is now known.

4.3.6 Pose Refinement

The fact that a mosaic can be constructed means that image registration and pose determination are equivalent problems. Many approaches to a solution are possible. Anticipating future work, we have used a Kalman filter [17] which determines the planar pose which aligns a set of corresponding planar point features. While the assumption can be easily relaxed to an affine transform, if needed, we presently expect several features to move as a rigid unit. To compute the pose of the image, attach a model frame M to an arbitrary location on it. Similarly, attach a world frame W to an arbitrary location on the mosaic.

The predicted positions of the point features with respect to the model frame ${}^{m}\underline{r}$ come from their positions in the image. The observed positions of the corresponding features ${}^{w}\underline{r}$ come from their positions in the mosaic. The problem is to find the pose $\underline{\rho} = (a, b, \theta)^T$, or associated transform ${}^{w}_{m}T(\rho)$ which best aligns the corresponding points. The situation is summarized in Figure 17.

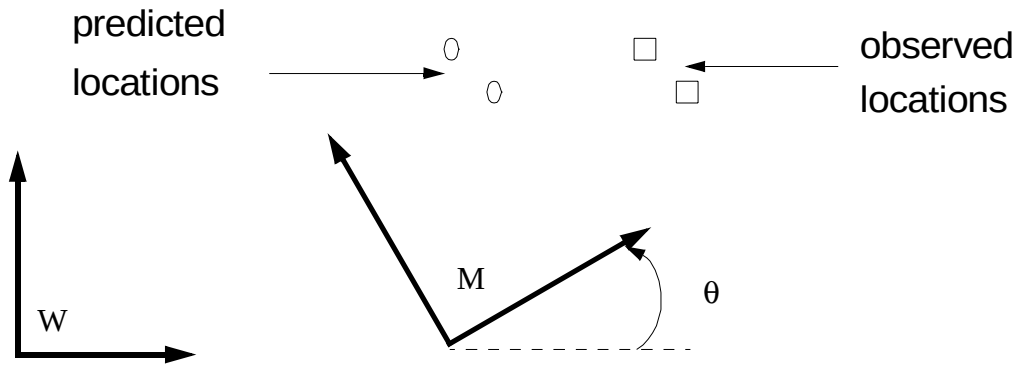

Figure 17: Observer Formulation. Given a set of point positions ${}^{m}\underline{r}$ expressed in the model frame M, and a corresponding set of point positions ${}^{w}\underline{r}$ expressed in the arbitrary frame W called "world", find the best pose of frame M with respect to frame W which brings the points most nearly into coincidence.

The prediction equation, or *observer*, tells us how to predict the locations of the points in the mosaic (world frame) from their locations in the image (model frame).

$$ {}^{w}\underline{r} = [{}^{w}_{m}T(\rho)]\,{}^{m}\underline{r} $$

If we denote the Kalman state vector thus:

$$ \underline{x} = (a, b, \theta)^{T} $$

then this relationship is of the form $\underline{z} = h(\underline{x})$ of the standard observer where the pose is the state vector, the model locations of the points are constant, and the prediction is the world locations of the points.

4.3.7 Output Transforms

The updated camera pose is now converted back to the time instant of the client system and the camera pose and mosaic tracking offsets are removed. Depending on the time instant to which the result corresponds, the client may now account for the motion that took place from then to the present.

4.4 Mosaic Builder

This section describes our current technique for constructing a mosaic. Globally consistent mosaics are constructed from segments in a multi-step process which:

- fuses images along individual edges in the network,
- resolves disagreements between them at intersections,
- stores the result in minimal memory.

Once the consistent mosaic is constructed, it can be used by all vehicles in the AGV fleet. Mosaic sharing is accomplished by rectifying each vehicle's imagery so that it produces the same viewing geometry as an ideally positioned pinhole camera.

While techniques for adjusting intensity levels to remove seams [6] and internally warp images into mosaics are well-known, we have not found them necessary in our work.

4.4.1 Global Consistency

Recall earlier comments to the effect that repeatability of localization requires a globally consistent mosaic. Others have addressed this problem successfully in an underwater mosaicking context [15]. A more operational definition of consistency, since a mosaic can never be perfect, is to require that the mosaic be everywhere “smooth” to some threshold.

Addressing this problem for large two-dimensional areas is a matter we have been able to avoid so far. However, we have found it necessary to address the global consistency of graph-like networks of otherwise one-dimensional mosaics.

The fundamental issue in mosaic fusion is a set of residual errors at intersections that must be resolved simultaneously. In the case that the mosaic network contains cycles, it is likely that no rigid transformation of map segments will resolve the errors - some amount of warping will be necessary.

4.4.2 Resolving Inconsistency

Each piece of mosaic associated with an edge of the guidepath network is initially collected as a separate map segment. Each collected image has an associated pose derived from the position estimate available at the time the image was acquired. In practice, such segments are internally consistent to a pixel at the time of construction due to the quality of odometry.

However, the errors that arise where segments intersect are due to odometry errors accumulated over perhaps hundreds of image widths, so these errors can be significant. The simultaneous solution of the residual equations that could be written for each intersection would be an ideal method of solution.

Our method is similiar to the frame-to-mosaic scheme of [37]. It is admittedly inferior to simultaneous solution but has been more expedient in practice. We create a mosaic by *sequentially* placing each segment into the right place in a global frame of reference. Once a segment is placed, it is not moved and the movements and distortions necessary to resolve intersections must be borne entirely by incoming segments.

In the case of cyclic networks, warping the internal structure of a segment is performed in order to ensure that endpoints rectify. Consistency is achieved through establishing correspondences between identifiable common features in the imagery of two or more intersecting segments.

The results are merely self-consistent if the ultimate location of a feature is determined from its unwarped location the first time it was added to the mosaic. However, calibration to an external standard, such as a CAD drawing, is also possible if the ultimate location of a feature is forced, through warping, to agree with the standard when the feature first enters the mosaic. In either case, all other segments are thereafter forced to agree with the initially assigned location.

In the self-consistent case, because locations are determined from odometry, later segments may be required to bear increasing distortion to account for the now irreversible errors in earlier segments. In the externally-consistent case, because such standards tend to be more self-consistent, less warping may be necessary.

In either case, provided the mosaic is sufficiently smooth, it becomes trackable and repeatable, although its accuracy is entirely determined by matters of calibration.

4.4.3 Mosaic Construction Example

Figure 18 visually indicates various intermediate stages in the process of constructing a globally consistent mosaic. A single segment is shown containing many calibration features, indicated by black x's along its length. The segment configuration in figure a is determined by odometry. In figure d, the segment has been rendered globally consistent at the two places where it folds on itself.

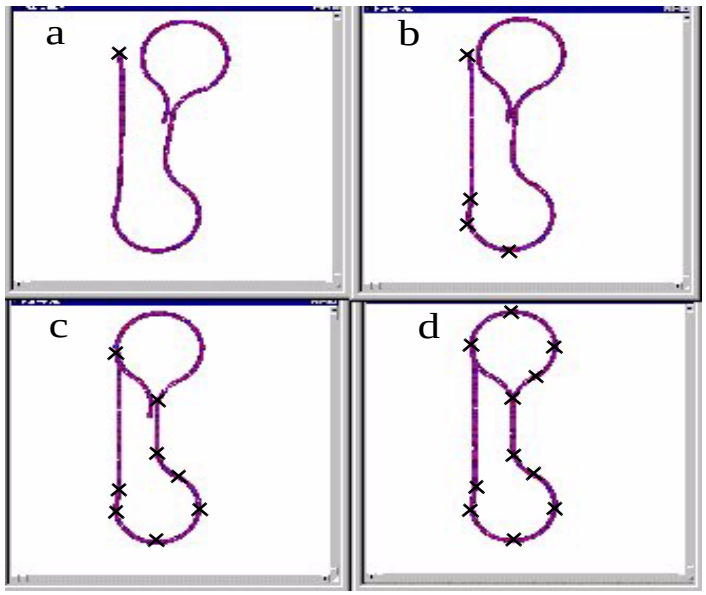

Figure 18: Construction of a Globally Consistent Mosaic. Segments are sequentially forced to agree at the calibration featues indicated by the black x's. Where the mosaic segments close on themselves, segments are distorted as necessary to remove odometry error.

4.4.4 Map Segment Condensation

Map segments have been represented as ordered sets of localized images up to this point because the last two operations required one-dimensional, ordered structure. At this point however, all images in all segments have been assigned their final poses and the extra memory required to store overlapping images has become wasteful

Each map segment is now condensed into one large image as shown in the following figure. A simple algorithm is used where pixel values in regions of overlap are taken from the component image whose center is closest to the pixel location. Memory reductions to 25% of original size are typical.

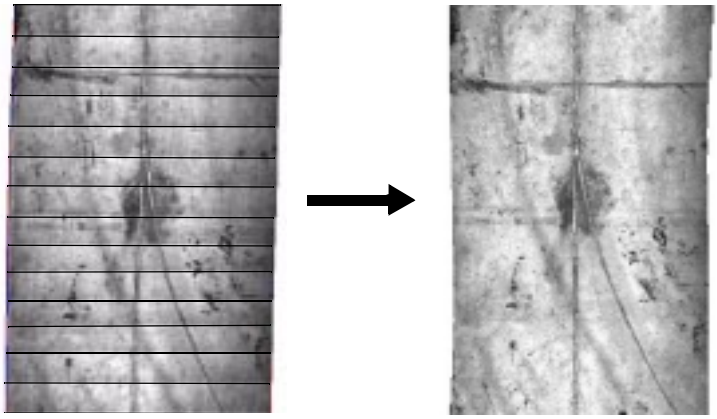

Figure 19: Segment Image Fusion. All images in a segment are fused into a single image.

5. Implementation and Results

This section summarizes our implementation of several mosaic-based localization systems and the results achieved.

5.1 Pragmas and Lessons Learned

It should be noted that the basic concept of mosaic-based localization can be applied to any vision based localization problem satisfying our formulation constraints. It could be used on microscopes imaging bacteria or telescopes imaging galaxies. Our experiences have been related to an application to AGV's. The issues arising in other applications may be very different than for AGV's.

5.1.1 Handheld Tests

Our initial demonstrations showed that a handheld camera manually scanned very slowly over carpeted floors can produce linear mosaics and then track camera motion over them. A fairly steady hand could mimic constant height and normal orientation reasonably well, and the algorithm was extremely robust to what subjectively appeared to be nearly repetitive texture in carpet patterns.

Speed was severely limited with no available estimate of inter-image motion. A dynamic model which estimated inter-image motions from the integration of linear and angular speed states increased speed significantly.

5.1.2 First Vehicle Mounted Tests

The next step was to move the system to a vehicle and add odometry. Several new issues emerged immediately. A first system was conveniently mounted on the rear of the vehicle for evaluation purposes.

As we expected, it suffered many false positives due to shadows cast by overhead lighting operating on the camera support structure (causing shadows stationary in the image) and on the structure in the building (causing shadows moving in the image).

It also became clear that increased distance of the camera from the odometry reference point amplifies the effect of both odometry errors and the slight pitching and rolling of the vehicle since both give rise to errors in predicted camera pose. Hence, the location of the camera relative to the reference point should be reasonably well calibrated.

5.1.3 Undercarriage Mounting

We immediately moved to a system mounted underneath the vehicle in order to control lighting. That step necessitated the use of a wide angle lens since field of view needed to be about 40 cm whereas camera height was limited to 40 cm. The lens made it necessary to warp images to remove lens distortion so that a region on the floor maintained its shape regardless of where it occurred in an image.

While lighting can be controlled under the vehicle, it turned out not to be so easy to do so. The need to fit under low undercarriage vehicles meant that there was very little space for a point source to diffuse enough to eliminate bright spots in imagery. Also, imagery of reflective surfaces contained little more than the reflection of the lights.

To solve these problems, we developed several sizes and versions of the highly engineered custom lighting and imaging module illustrated in Figure 20:.

Figure 20: Lighting and Imaging Module. This device is a custom solution for low power, engineered lighting. Its slimline form factor renders it mountable underneath vehicles. Resulting light levels are uniform over the floor.

Due to this device, it has become unnecessary to process mosaics for lighting variations on many floors. Of course, the reason for using the system is to generate sufficient illumination in regions which were otherwise too dark to extract texture.

Processing performance is highly related to some matters of geometry. The highest level at which an imaging module could be mounted was about 8 inches at the image plane. Also, we chose to use a pixel size of 2.5 mm in order to generate about 100 pixels across an image, and updated lookup tables appropriately.

5.1.4 Floor Texture Tests

Floor texture has been an issue from time to time on some floors. Often, newly painted floors completely lack visible texture. Rather than working at the time on several ideas to raise the texture sensitivity of the system under these circumstances, we elected to paint random texture on the floor in such areas. We also encountered one case where texture on a metal ramp was highly repetitive on a one inch scale and painted the area for the same reasons.

Such problems can also be resolved without such methods by running on other navigation sensors (odometry, range sensors) over such regions when they cannot be avoided in more direct ways.

5.1.5 Initial Mosaic Building - Dirt and Fiducials

Initial mosaic building work was simplified by deliberately placing high contrast, easy to distinguish marks on the floor at the center of intersections. These features were designed to be

easy for the operator to find when running the map construction software.

Experiments on our first large map, over a period of many months, showed that it is best to map a floor when it is relatively clean. In practice, it seems that recording the transient (dirty) component of the appearance of surfaces is not a good idea. However, we noted that that the system worked flawlessly on highly-trafficked floors that were not cleaned for months.

5.1.6 Floor Geometry Tests

On occasion, such as on ramps leading between different areas, the assumption of entirely flat floors has been violated. It is easy to augment our algorithms to relieve the assumption of a normal, constant height camera by adding three new camera pose states, and it is easy to therefore relax our flat world assumption to a locally flat one. Both come, however, at the cost of extra processing in all cycles, so we have avoided this solution.

The greatest practical difficulty with ramps is the fact that camera attitude depends on where the wheels fall on the floor. Unless the camera is symmetrically mounted between the wheels, it has a different off-normal orientation when the vehicle travels forward and backward over the same spot on uneven floors. We have found it expedient to simply run on odometry until ramp transitions are over and to ignore the fact that it introduces motion in the vertical direction.

5.1.7 Mosaic Sharing

Subsequent attempts to operate several vehicles simultaneously in the same large test area brought several more issues to light. It now became preferable for a single common mosaic to be used by all vehicles. Mosaic sharing raised issues of having two independent imaging systems precisely agree on appearance. This lead to the need for levels of consistency beyond the obviously needed agreement between vision and odometry.

Our solution was to reapply the lens distortion lookup tables to compensate for camera pose errors as well. We bypassed external camera calibration parameters in favor of direct image rectification, a technique used in stereo, because camera pose itself was not needed and template matching required rectification (as well as normalization) anyway.

Basically, an image of a calibration grid was forced to appear to be a perfectly symmetric grid of perfect size. In this way, we forced all cameras on all vehicles to produce the image expected of an ideal pinhole camera that was optimally mounted. A typical rectification pair is shown below.

5.1.8 Pose Tracking in Large Scale Mosaics

When trying to operate over very large distances, it became apparent that we could not and should not rely on agreement between odometry and vision over large excursions. However, it was useful to use optimal filtering techniques in order to fuse the two. Doing both of these things simultaneously required a slight reformulation.

The optimal estimate of camera pose is not the appropriate place in the mosaic to start a search for template matches because it could rightfully drift many search radii beyond the true positions of the templates. A separate mosaic tracking pose for the camera needs to be maintained by successively adding differential optimally-estimated (filtered) updates to the last mosaic-based

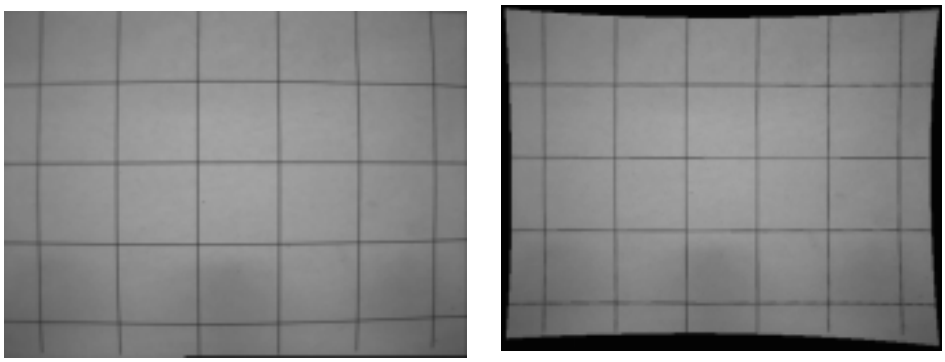

Figure 21: Image Recification. The left image is the raw input from the camera. The right has been forced to agree with an ideal pinhole camera at each grid intersection.

(unfiltered) pose.

5.1.9 Cycles in Mosaics

We had managed to avoid the issues of mosaics containing cycles for a long time by operating on mosaics whose topology was a tree. For such mosaics, it is possible to traverse the tree from the root and rigidly reassign the pose of any subtree in order to remove slight registration errors at the intersections.

Once a mosaic contains cycles, nodes may not be visited only once in a traversal and fixing an error in one place may aggravate one elsewhere. Sub-mosaics could no longer be moved rigidly, so our mosaic warping/smoothing algorithms were then introduced.

5.1.10 Mapping Rig

A subtler way in which cycles appear in maps occurs when they are wider than a single image. Our initial vehicle was designed for manual use so its design made no provision for vision systems attached to its underside. The camera had to be mounted well off-center and this meant that substantially different regions of the floor were scanned when the vehicle travelled in opposite directions but precisely over the same path.

We avoided this issue by the introduction, for other reasons, of the ***mapping rig*** vehicle (shown below) because it generates maps as wide as the vehicle itself. The original motivation for the mapping rig was the need to build mosaics of remote facilities and test them before committing to ship a robot vehicle there. However, given the opportunity to design this vehicle more optimally for its purpose, we elected to image very wide floor swaths on the argument we could always reduce their width later.

The rig is equipped with a large version of our imaging system (see Figure below), odometry, and a computer with display and keyboard. It stores enough energy in its batteries for several hours of untethered operation.

5.1.11 Standalone Vision Localization Module

Once it became desirable to test the system on a commercial vehicle, it became necessary to extract it both logically and physically from a larger system. A separate process running on a separate

Figure 22: Mapping Rig. This custom engineered vehicle is used to create mosaics. It couples a downward looking camera for acquiring imagery with an odomery system for estimating relative motion. between images.

computer was needed.

It was desired to configure the system to take input position estimates from any source and update them to reflect the differences between those estimates and the vision-based estimate.

In doing so, it became immediately clear that one pixel of error (and hence one less pixel of useful search radius) could be caused by an unmodelled communication delay of as little as 1/2 millisecond between the two involved computers. It became necessary to employ measures to reduce and then model any remaining such delays. This issue is fundamental to the fact that the ratio of system resolution to its own operating velocity (the theoretically minimum significant delay) is 1/2 millisecond.

5.1.12 Installation Calibration Standard

When installing a system in its first field test site, a matter of standards became a clear issue. It is common in AGV installations to design the guidepath network before the vehicles are shipped to the site. This is done based on a model of the building that is either specifically generated for that purpose by a survey or extracted from a preexisting CAD model.

In such a case, the guidepath network is effectively off-line programmed rather than taught. While it is still very repeatable, it is likely that the map building process and the CAD model or survey will disagree significantly on the location of points on the floor. This disagreement may in turn mean that a CAD referenced guidepath, or at least a vehicle trying to follow it, may have its camera entirely leave the map.

The converse dependency is also an issue. It is convenient for the mosaic to be constructed before

vehicles are deployed - at least because the mosaic can be extensively tested off-line. However, in this case, the map building camera must be driven over the guidepaths and it has not been straightforward in real settings to efficiently guarantee that the correct regions of the site floor have, in fact, been imaged. Surveying the guidepaths beforehand is, of course, feasible but expensive.

5.1.13 Vehicles, Hardware, and Test Sites

One mosaic-based localization system for our 40,000 square foot test facility has been in operation for four years. Systems have also been deployed in two other facilities. It has been used on forked, tug, and unit-load AGVs. The tug AGV is shown in Figure 23:.

Figure 23: Tug AGV. This automated guided vehicle is one of several with mosaic-based positioning installed.

Variants have also been applied by others to mining vehicles imaging the walls of mines.

5.2 Performance

The performance of the system against some of the metrics derived earlier is presented here.

5.2.1 Memory Usage

To date, four mosaics have been constructed and used. These mosaics have ranged in size from 80 to 200 linear meters - all with closed loops. These mosaics have been small enough to store in RAM.

5.2.2 Texture Tolerance

We have observed excellent noise immunity in the correlation algorithm. It operates robustly in the

face of months of cumulative dust and grime which can hide the underlying floor texture that was originally mapped. Concrete floors with exposed aggregate work especially well. We have also verified outdoor operation over textured ashphalt over time periods including intervening inclement weather.

5.2.3 Processing Performance

Experiments have shown that a 800 MHz Pentium III can perform 1.5 X 10^8 correlation operations per second, when use of the MMX instruction set is forced. Based on this value, Table 3 summarizes the performance of this CPU for the indicated template and search parameter values.

Table 3: Correlation Speed of Pentium III

Symbol	Value	Symbol	Value
f_{corr}	150 Mops/ sec	w	11 pixels
f_{templ}	1.2 Mops/ sec	N	4
f_{search}	300 Kops/ sec		

On the basis of Figure 9, and the above result, a vehicle speed of 30 m/s can be supported at 0.1 seconds cycle time. We have at times observed reliable tracking under good calibration as fast as a forktruck can be comfortably driven in a straight line, (perhaps 8 m/s) using a Pentium II, but we have not attempted the above highway speed figure on the Pentium III. Our normal AGV operating speed is a mere 2.5 m/s which corresponds to a “mere” 1000 times resolution. We redirect the computational capacity to drive fast to enhancing reliability at slower speeds by overconstraining pose refinement and using very large templates and search regions.

5.2.4 Mapping Speed

The overlap speed limit applies to the process of on-line mosaic construction, but not to mosaic tracking. For an overlap of 30%, and our image size of 0.3 meters, and 10 Hz update, this gives 2.1 m/s as the maximum vehicle speed sustainable while automatically building mosaics.

If mosaics are constructed on-line, there is a clear motivation to decrease cycle time as much as possible. If sufficient cycle time reduction is not possible, images can be stored at frame rate and processed off-line, as long as they still overlap.

5.2.5 Safe Distance after Loss of Visual Lock

Under the assumptions leading to Figure 8, for a typical search radius of 12 pixels = 3 cm, the processor’s ability to search the required radius is exceeded in 35 cm of motion since the last fix. In practice, it is possible to simply stop when this failure occurs and conduct a wider search of the mosaic. If the loss of visual lock was not due to significant appearance changes, or excursion off the mosaic, the tracker error can be resolved and the system can continue to operate.

5.2.6 Installation Time

Building mosaics is a process which requires that the entire network of guidepaths be walked at least once at normal human walking speed (approximately 0.5 m/s). Thereafter, off-line mosaic registration requires a few minutes per intersection. Extrapolating from our experience, total mosaic creation for a large site, from the initial preparatory steps, to the image collection, to the final off-line creation, can be done in less than a day.

5.2.7 Excursion

To date, four maps have been constructed and used. These maps have ranged in size from 80 to 200 linear meters, some containing large arcs, and all with closed loops.

In our local test facility, a network of guidepaths has been mapped and rendered geometrically consistent for our testing purposes. This map is shown below. The short vertical segments of the map are areas where the vehicle interfaces with racks and loads.

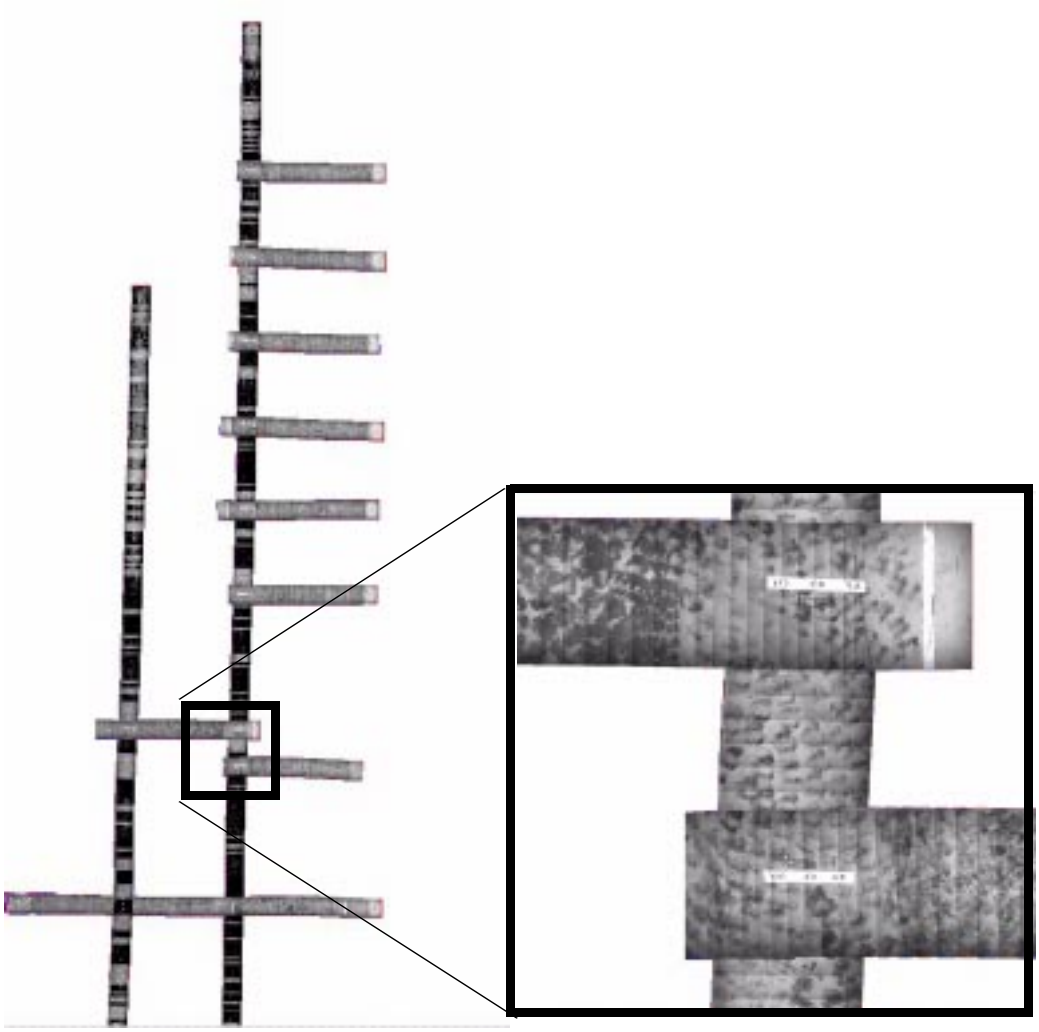

Figure 24: Network Mosaic and Exploded View of Component Imagery. This network of images covers part of our test facility floor.

5.2.8 Repeatability and Accuracy

On our mosaic at CMU, the system has operated for four years producing 1 mm repeatability at speeds occasionally exceeding 15 mph - more than safety regulations would allow outside our controlled laboratory setting. The 1 mm repeatability stems from the choice of 2.5 mm pixel size and the fact that correlation surfaces exhibit peaks sharp enough to permit subpixel template localization. Individual processing cycles where no fix can be computed are virtually nonexistent. Generally, absolute accuracy has not been a concern in our tests, so it is not quantified - nor is it particularly relevant.

5.2.9 Reliability

It has been our experience that mosaic-based positioning is a very practical, high performance, infrastructure free solution to vehicle guidance in simply structured environments. Results have shown it to be a competitive position estimation technique for at least industrial AGVs and perhaps for other vehicles. We have run vehicles in our facility that do not get lost for weeks of sustained operation until someone decides to change the software and “improve” things. Recently, we have executed a 40 hour endurance qualification test. Some test details are summarized in Table 4.

Table 4: Results of AGV Qualification Test

Parameter	Result
Total Elapsed Time	13 days
Total System On Time	97.5 hours
Total Distance Travelled	107.8 miles
Average Speed	2 mph
Images processed	838,887
Total Loss of Lock Events	3

The table provides detailed records of the last 13 days of an 8 week test program. Reliability continued to improve over this 8 week period as problems were diagnosed and fixed. The difference between system on time and elapsed time reflects the schedule of the part-time testing staff. Most continuous testing periods were from 4 to 8 hours. This result indicates that the system can operate for an entire week before an event (such as unrecoverable loss of visual lock on the mosaic) requiring human intervention occurs. Such human intervention would typically last only a few minutes while the system was re-registered on the mosaic.

6. Summary and Conclusions

6.1 Summary

This article has discussed an important new approach to mobile robot and vehicle localization based on building and tracking mosaics. Such an approach to localization enables infrastructure-free, free-ranging vehicle guidance exhibiting repeatability everywhere in the target environment on the order of a pixel. Even large scale mosaics can be built relatively easily and cheaply with contemporary technology. Our restriction to substantially flat environments, while theoretically strict, excludes only a relatively small percentage of all wheeled vehicles in operation today. Many such environments also exhibit persistent visual texture.

6.2 Motivation

Although the technique of using photorealistic scene models might be considered extreme, such models can be:

- *necessary* due to the sparsity of appearance features on many man-made surfaces,
- *viable* due to validity of several simplifying assumptions,
- *useful* due to superior performance relative to alternatives.

If one commits the effort required to construct a mosaic, the benefits can make the time well invested.

6.3 Feasibility

We have also found that many floors do exhibit enough visual texture for mosaic-based navigation to work. It may have sounded absurd even a decade ago to propose navigating from a huge 10 Gbyte image of a factory floor. However, we have now reached the point where memory, processing, and imaging hardware are both adequate and inexpensive. Consider that:

- A contemporary double-sided, double layer DVD-18 ROM disk can store 17.0 GBytes. This represents, at 1 cm resolution, a swath of imagery 1 ft wide from Boston to Seattle - completely across the continent of North America. Many applications can store usefully large areas in 100 MByte solid state flash disks - similar to those used in consumer digital cameras.
- Real-time vision applications that required dedicated specialized computers (e.g. dense stereo) are now routinely performed on desktop computers. The Intel Pentium MMX extensions perform template correlation in a single instruction.
- CMOS cameras can cost as little as $80 and digitizers $100.

6.4 Performance

Two important fundamental figures of merit that characterize overall performance are excursion and velocity expressed in units of system resolution. For mosaic-based localization, the first metric becomes bounded only by off-line memory and guidepaths 20 units wide and roughly 1 billion units long can be represented. Speeds in excess of 1000 units per second can be tracked visually. Stating results in this way permits extrapolation to aerial, space, and other applications.

6.5 Applications

An application to automated guided vehicles has been presented, whereas the technique applies to the problem of determining motion of a camera over any scene that can be represented by a mosaic. Designs for microscopic and planetary scales are equally possible since imagery can be produced

at such scales. While none of the assumptions of a completely known scene, or restricted camera motion, or externally estimated motion are necessary in general, a field-relevant result can be achieved when those assumptions are justified and exploited.

Some applications for which this approach to localization is likely to be beneficial include:

- service robots in homes, offices, and facilities performing such duties as food and mail delivery, waste disposal, entertainment, surveillance, and floor care,
- container and material handling in paved outdoor rail, trucking, and shipping yards,
- small materials handling in factories and order picking in warehouses.

6.6 Acknowledgements

This work was funded in part by Ford Motor Company. Results described here have been achieved through the support of many dedicated individuals including Mark Ollis, Laura Cassenti, Dave Stager, Greg Podnar, Yu Zhong, Byan Nagy, Ethan Frantz, Mike Happold, and Patrick Rowe.

7. References

[1] C. S. Andersen, S. Jones, J. Crowley, “Appearance Based Processes for Visual Navigation”, Proc SIRS’97, pp227-236, 1997.

[2] 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.

[3] J. Borenstein, B. Everett, and L. Feng, "Navigating Mobile Robots: Systems and Techniques." A. K. Peters, Ltd., Wellesley, 1996.

[4] W. Burgard, D. Fox, Daniel Hennig, and Timo Schmidt, “Estimating the absolute position of a mobile robot using position probability grids,” In Proc of the Fourteenth National Conference on Artificial Intelligence (AAAI-96), pages 896-901, 1996.

[5] P. J. Burt, C. Yen, and X. Xu, “Local correlation measures for motion analysis: a comparative study.” IEEE CPRIP, 269-274, 1982.

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

[7] R. Chatila and J. P. Laumond, “Position referencing and consistent world modelling for mobile robots,” in Proc. IEEE Int. Conf. on Rob. and Aut. St. Louis, MO, Mar 1985, pp 138-145.

[8] R. Chatila, G. Andrade, S. Lacroix, and A Mallet, “Motion Control for a Planetary Rover”, in Proc 1999 International Conference on Field and Service Robots”, Aug 1999, Pittsburgh, PA.

[9] E. Chen and L. Williams. “View Interpolation for Image Synthesis”, In Proc. SIGGRAPH 1993.

[10] J. L. Crowley, “World modelling and position estimation for a mobile robot using ultrasonic ranging”, Proc IEEE Int. Conf. on Rob and Aut., 1989.

[11] C. Drocourt, L. Delahoche, C. Pegard, and A. Clerentin, Probablistic Localization by Appearance Models and Active Vision. Proc IEEE Int. Conf. on Rob. and Aut., 1999.

[12] H. F. Durrant-Whyte, “The design of a radar-based Navigation System for Large Outdoor Vehicles”, Proc IEEE Int. Conf. on Robotics and Automation, Aichi, Japan, pp 764-769.

[13] T. Einsele, “Real-Time Self-Localization in Unknown Indoor Environment Using a Panorama Rangefinder.”, In IEEE/RSJ International Workshop on Robots ans Systems, IROS 97.

[14] A. Elfes, Sonar-based real world mapping and navigation, IEEE Journal of Robotics and Automation, Ra-3(3), 1987.

[15] S. Fleischer, S. M. Rock, and R. Burton, Global Determination and Vehicle Path Estimation from a Vision Sensor for Real-Time Video Mosaicking and Navigation, Ocean, Vol 1, 1997

[16] D. B. Gennery. Visual tracking of known three-dimensional objects. *Int’l Journal of Computer Vision*, 7(3):243-270, 1992.

[17] A. Gelb, Ed., Applied Optimal Estimation, MIT Press, Cambridge MA, 1974.

[18] J. Guivant, E. Nebot, and S. Baiker, “Autonomous navigation and map building using laser range sensors in outdoor applications,” to appear in Journal of Robotic Systems.

[19] G. Hager, and P. Bellhumeur. Efficient Region Tracking with parametric models of illumination and geometry. *IEEE Journal of Patt. Recog and Machine Intelligence* 20(10),1025-1039, Oct 1998.

[20] R. Haralick, H. Joo, C. N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, “Pose Estimation from Corresponding Point Data. IEEE Transactions on Systems, Man, and Cybernetics, 19(6):1426-1446, Nov/Dec 1989.

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

[22] R. Horaud, B. Conio, O Leboulleux, and B. Lacolle, An Analytic solution for the perspective 4-point problem. Computer Vision, Graphics, and Image Processing, 47(1),:33-44, 1989.

[23] M. Irani, P. Anandan, and S. Hsu. mosaic-based representations of video sequences and their applications. *Proc. Intl. Conf. on Computer Vision*, pages 605-611, 1995.

[24] A. Kelly, “Some Useful Results for Closed Form Propagation of Errors in Vehicle Odometry”, CMU Tech report CMU-RI-TR-00-20.

[25] E. Krotkov, Mobile Robot Localization from a Single Image, *In Proc IEEE Int Conf. on Rob. and Aut*., pp 978-983, 1989.

[26] B. J. Kuipers and Y. Byun, A robust, qualitative method for robot spatial learning, In Proc 1988 Nat Conference on Artificial Intelligence, vol 2, pp 774-779, 1988.

[27] S. Laveau and O. Faugeras, “3D Scene Representation as a Collection of Images”, in Proc. ICPR, 1994.

[28] J. J. Leonard and H. F. Durrant-Whyte. “Simultaneous Map Building and Localization For An Autonomous Mobile Robot”, *In IEEE/RSJ International Workshop on Robots and Systems, IROS 91*, Pages 1442-1447, 1991.

[29] J. Leonard and H. Durrant-Whyte, Mobile Robot Localization by Tracking Geometric Beacons, IEEE Trans on Rob. and Aut. Vol 7, pp 376-382, 1991.

[30] F Lu, and E. Milios, Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans, Journal of Intelligent and Robotic Systems, 18:249-275, 1997.

[31] B. D. Lucas and T. Kanade. “An iterative image registration technique with an Application to stereo vision. In proc. In.t Joint Conf. on Artificial intelligence, pages 674-679, 1981.

[32] L. Matthies, T. Kanade, and R. Szeliski. Kalman filter based algorithms for estimating depth from image sequences. International Journal of Computer Vision, 3(3):209-236, September 1989.

[33] J. Niera, J. D. Tardos, J. Horn, and G. Schmidt, “Fusing Range and Intensity Images for Mobile Robot Localization”, IEEE Trans on Rob. and Aut., Vol 15, No 1, p 76, 1999.

[34] C. Olson, L. Matthies, M. Schoppers, M. Maimone “Robust Stereo Ego-Motion for Long Distance Navigation”, in Proc. Conference on Computer Vision and Pattern Recognition (CVPR ‘00).

[35] C. Olson, Mobile Robot Self-Localization by Iconic Matching of Range Maps, In Proc of 8th Intl Conf. on Advanced Robotics, pp 447-452, 1997.

[36] H. S. Sawhney, S. Hsu and R. Kumar, Robust video mosaicking through topology inference and local to global alignment, In Proc. European Conference on Computer Vision, ECCV, Freiburg Germany vol 2, pages 103-119, June 1998

[37] H. S. Sawhney, R. Kumar, G. Gendel, J. Bergen, D. Dixon, V. Paragano, VideoBrush: Experiences with Consumer Video Mosaicking, Fourth IEEE Workshop on Applications of Computer Vision, WACV, Oct 98

[38] M. Schmitt, M. Rous, A. Matsikis, K. F. Kraiss, “Vision based self-localization of a mobile robot using a virtual environment”, Proc IEEE Int. Conf. on Rob. and Aut., Detroit, MI, May 1999.

[39] J. Shi, and C. Tomasi, “Good features to track”, In Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR94), Seattle, June 1994..

[40] B. Schunck, “Image flow: Fundamentals and future research,”, Proc IEEE Conf. Computer Vision Pattern Recognition, San Francisco, 560-571, 1985.

[41] Richard L. Sparks, Stephen M. Rock, and Michael J. Lee, Real-Time Video Mosaicking of the Ocean Floor. *IEEE Journal of Oceanic Engineering*, Vol 20, No. 3, July 1995.

[42] R. Szeliski. Image mosaicking for tele-reality applications. IEEE Wkshp. on Applications of Computer Vision, pages 44-53, 1994.

[43] R. Talluri and J. K. Aggarwal, “Position Estimation Techniques for an Autonomous Mobile Robot - a Review., Handbook of Pattern Rcognition and Computer Vision, pp 769-801, World Scientific.

[44] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization Method. Int J. Computer Vision, 9(2):137-154, 1992.

[45] S. Ullman, The interpretation of visual motion, MIT Press, Cambridge Ma, 1979.

[46] M. Vincze, “Optimal window size for visual tracking”, In Proc ICPR, 1996.

[47] M. Vincze, and C. Weiman “On Optimising Tracking Performance for Visual Servoing”, In Proc ICRA, 1997.

[48] J. Weng, T. Huang, and N. Ahuja, “3D motion estimation, understanding, and prediction from noisy image sequences”, IEEE Trans Pattern Analysis and machine intelligence, 9(3), p 370-389, May 1987.

[49] T. Werner, R. D. Hersch, and V. Hlavac, “Rendering Real-World Objects Using View Interpolation”. In Proc ICCV, Boston, 1995.

[50] L. Wixon, J. Eledath, M. Hansen, R. Mandelbaum, D. Mishra, Image Alignment for Precise Camera Fixation and Aim, Proc. Conference on Computer Vision and Pattern Recognition (CVPR ‘98).

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