Position Estimator for Underground Mine Equipment

Gary K. Shaffer, Anthony Stentz, William L. Whittaker, and Kerien W. Fitzpatrick

Abstract—**This paper describes a 2-D perception system that exploits the accuracy and resolution of a laser range sensor to determine the position and orientation of a mobile robot in a mine environment. The perception system detects features from range sensor data and matches the features to a map of the mine to compute the sensor position. The features used are line segments and corners, which represent the typical geometry of the mine walls and intersections found in room-and-pillar type mining. The position estimate is refined by minimizing the error between the map and sensed features. This position information can be used for autonomous navigation when a map of the mine is available or to survey the mine to build such a map. The technique is applied to robot navigation in a mine mockup. A refinement of this system could guide machines to yield productive, safe mining operations.**

I. Introduction

UNDERGROUND mining is a cooperative enterprise of powerful, mobile equipment and the workers who operate it. If mining equipment could be automated to function without a worker's full attention, the mining industry could enhance productivity, access "unworkable" mineral seams, and reduce human exposure to the inhospitable environment of dust, noise, gas, water, moving equipment, and roof fall. The critical missing link to enable mine automation is the capability of equipment to estimate its position relative to its surroundings (self-location). (Note that we often use the word "position" to refer to combined position and orientation). Several methods of self-location such as dead reckoning, inertial systems, beacons, and guidewires fail to satisfy the demands of the mine environment. There is a clear, compelling need for a position estimator relevant to robotic mine equipment. This paper develops and demonstrates, for the first time, a viable technology for self-location by mine equipment.

Our approach is feature based and uses both line segments and corners. It can handle the expected uncertainty in the initial estimate (heavy clutter in the environment) and still compute an estimate quickly.

A review of background and related work appears in Section II; position estimation algorithms and software are described in Section III; a physical robot implementation and position estimation results are presented in Section IV. Section V is the conclusion.

Paper PID 91-42, approved by the Mining Industry Committee of the IEEE Industry Applications Society for presentation at the 1990 Tenth West Virginia University Mining Electrotechnology Conference, Morgantown, WV, July 24–27. This work was supported by Carnegie Mellon University under U.S. Bureau of Mines Contract HO 358021. Manuscript released for publication November 27, 1991.

The authors are with the Field Robotics Center, Carnegie Mellon University, Pittsburgh, PA 15213.

IEEE Log Number 9201853.

II. Background

Perception techniques gather data about the surrounding environment and infer position by interpreting relationship to the scene. Existing approaches to position estimation fall short of mine navigation needs. Explicit techniques estimate positions from sensors like beacons, gyroscopes, and wheel counters that measure position directly.

A. Explicit Position Estimators

The simplest position estimator for mobile equipment is dead reckoning, in which the robot estimates its current position by step counting (integrating its combined steering and propulsion history). This approach is vulnerable to bad calibration, imperfect wheel contact, upsetting events, and it provides, at best, only a rough position estimate. This estimate generally gets worse as the distance traveled (i.e., the length of the integral) increases.

Inertial navigation system (INS) position estimators use multiple gyros (mechanical or ring laser) and accelerometers (one for each axis) to provide an acceleration history that is integrated to estimate position. Although an INS is generally more accurate than dead reckoning, an INS is subject to gyro drift, calibration problems, and sensitivity limitations.

Sensors that can accurately locate beacons can estimate the absolute position of the robot from known beacon locations (Case [3], Cao [2], Giralt [8]). Major drawbacks are that beacons must be emplaced and within range, which means that a robot's worksite must be appropriately configured.

B. Position from Perception

Model-based perception can register sensed data to a world model to determine a robot's position. This method does not suffer from the integration errors inherent to dead reckoning and INS methods; however, the accuracy of the position estimate depends on the sensor and the accuracy and validity of the world model. This comparison of sensed data to model data can be accomplished in a number of ways. Two general paradigms are feature based and iconic. The feature-based method processes the raw data into a small set of features (such as line segments)

and matches these against the corresponding features in the model. Drumheller [5], Krotkov [10], and Crowley [4] used feature-based approaches. Drumheller used a range sensor to find line segment features to be matched against a world model to determine the position of a robot within a room (i.e., an environment that consisted of planar surfaces such as walls). Krotkov used a CCD camera to accurately determine the angles to vertical edges within the scene (from walls and cabinets). Given these angles and a map of the actual positions of the vertical edges, a position estimate was computed. Crowley used a range sensor to find line segments for constructing a world model and matching against it. Bloom [1] suggested using detailed descriptions of naturally occurring landmarks to enable a vision system to locate them within the scene.

Elfes [6] used an iconic matching technique developed by Moravec [11] to match *occupancy maps* in order to find a position estimate. Occupancy maps model the environment using a spatial grid of cells, where each cell has a probability value assigned to it, which represents the likelihood that the cell is occupied. The iconic method differs from feature matching in that raw data is matched directly to the world model.

The choice of technique depends on the nature of the problem. The feature-based approach reduces the data to a small set of features, enabling the system to try *all* matches of sensor to model features and, thus, compute the position with little or no initial estimate. However, poor quality features or too few features cripple the accuracy of the position estimate. Another disadvantage is that extracting features from the raw data takes extra time. The iconic approach has the advantage of using more data directly in the calculation of the position estimate; therefore, the accuracy of a position estimate is generally better. Iconic matches succeed without the need for explicit features in the environment. The disadvantage of the iconic approach is that the complexity of matching all raw data points to all model points is prohibitively high; consequently, a good initial estimate is required.

The type and quality of sensed data greatly impacts the choice of position estimation method. Many researchers have used wide-angle ultrasonic (sonar) sensors, often as a default, because of their low cost; however, the accuracy of these sensors is marginal, ambiguities occur, and the data is difficult to interpret correctly. Often, because of the texture of the reflecting surface or the angle of incidence of the sonar beam, a totally false range reading is reported. Furthermore, due to the large size of the sonar footprint, it is impossible to measure a scene with any detail. Relevant position estimation (especially when using a feature-based approach) is virtually impossible with wide-angle sonar sensors. Laser range sensors are more narrowly focused and provide much higher accuracy and ease of interpretation than sonar.

Another possibility is to process a video image to find features, as in Krotkov [10], Kak [9], Bloom [1], and others. Kak built a representation of the scene by segmenting the image into regions, finding the bounding edges of regions, and using them to define vertices, faces, and objects. This constructed scene representation was matched with an expected scene generated from a CAD database.

C. Mine-Specific Issues

The underground mine environment imposes several requirements on a position estimation system. Using dead reckoning alone is ruled out because of the irregular surface of the mine corridor floor, the often jerky motions of the mine machine, and especially, the large amount of slippage of the machine's tracks. Long-term autonomous operation of mine equipment precludes the use of an INS since errors from the gyro drift quickly exceed acceptable levels. It is desirable to work in newly mapped areas without first installing the necessary beacons (especially if the mapping is done automatically in conjunction with the position estimation). Furthermore, since the environment is cluttered with machines and people not in the model, a significant number of optical beacons could be occluded. Therefore, optical beacons alone are not viable in an underground mine environment. Because mining equipment is mobile, a relevant position estimator must have a reasonably short cycle time. The jerky motions of tracked mining machines and the subsequently poor dead reckoning preclude good position estimates by simple means.

All existing approaches fall short of mine navigation needs. Beacons will not always be within range. Jerky motions frustrate iconic methods. Existing feature-based approaches that use *only* lines or *only* corners require good initial estimates or take too much time matching against a complete map.

III. Perception System

A. Overview

The position estimator takes as input a model of the environment and a scan from the laser range sensor and produces an estimate of the robot's position. The laser range sensor measures distance to points in the robot's surroundings by rotating like a lighthouse and casting range measurements with its light beam. A scan is a dense array of range measurements taken at equally spaced angular intervals around 360° in a horizontal plane. Essentially, the scan is a planar profile of the sensor's environment formed by finding the distance to the nearest objects at a particular height. Position estimation is accomplished by predicting visible features from a map, extracting features from the scan, matching features from the scan and map, and estimating position from the match (see Fig. 1). Feature prediction generates a set of candidate visible features from an approximate position of the robot. Feature extraction finds features in the range scan. Features extracted from the scan are paired with features from the map in the process of feature matching. The position estimator finds a position that minimizes the error between sensed features and map features. These processing steps are described in detail below.

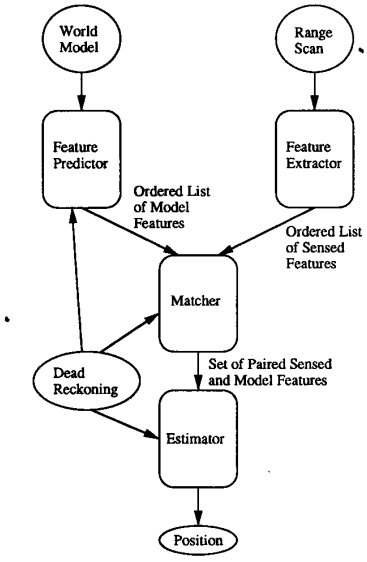


Fig. 1. Overview of position estimation.

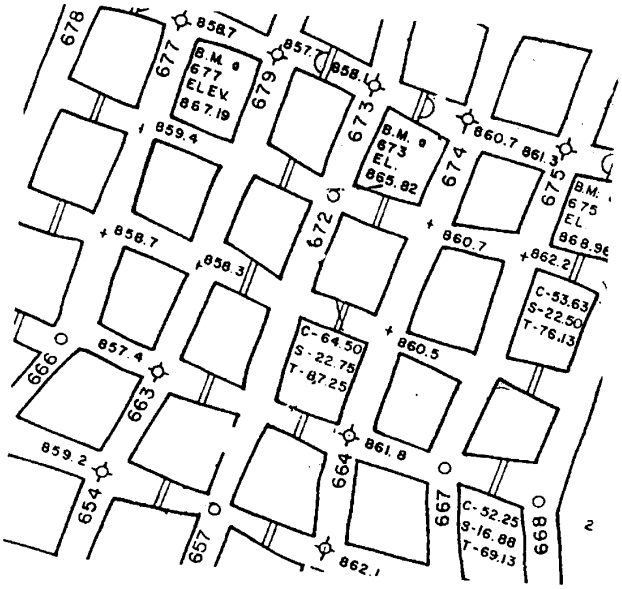


Fig. 2. Typical underground mine map.

B. World Model

A typical underground coal mine map appears in Fig. 2. For underground coal mines, this assumes that floors are flat and walls are nearly vertical. With these assumptions, a 2-D (line segment) model sufficiently represents the features of underground coal mines.

C. Features

A world model composed of line segments contains two categories of primitive features—line segments and corners. The term *corner* is used loosely to mean either a nonoccluding corner (formed by two visible line segments sharing a common endpoint) or an occluding corner (formed when a visible endpoint of a line segment obstructs the view of part of another line segment). These features are illustrated in Fig. 3. A corner is described by four parameters: x, y, *concave angle*, and *bisector orientation*. The concave angle is the smaller of the two angles formed by a corner's two line segments. The bisector orientation is the angle that the bisector of the concave angle forms with respect to the positive x axis.

Line segments are described by three parameters: length, perpendicular distance from the origin, and orientation, which is the angle that the perpendicular forms with respect to the positive x axis.

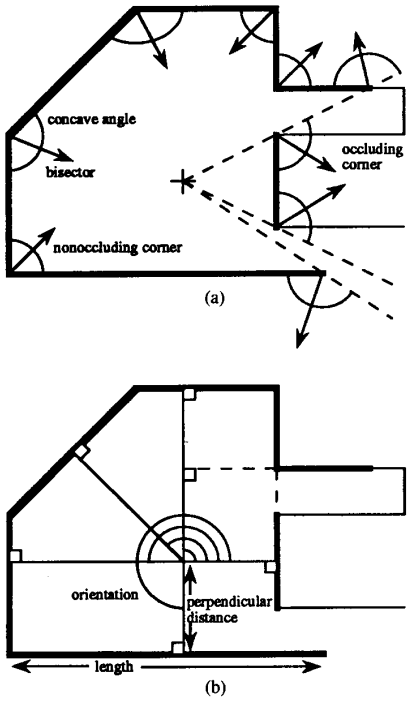


Fig. 3. (a) Corner; (b) line segments.

D. Feature Prediction

Since only a small portion of the model (in general) is visible from a particular location (x, y) of the robot, feature prediction greatly simplifies the process of feature matching (described below) by reducing the number of model features that can possibly match the sensed feature (see Fig. 4). The feature predictor assembles an ordered list of model features visible from a given (x, y) as the scanner sweeps in a counterclockwise direction. This preprocessing greatly constrains the possible combinations of pairings of sensed features and model features, thereby simplifying the feature matching process.

The feature prediction algorithm first creates a sorted array of the endpoints of all the line segments in the

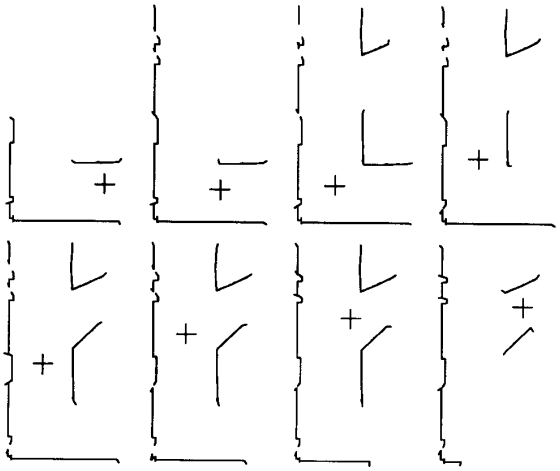


Fig. 4. Depending on the location of the robot (marked with a cross), the set of visible model features varies.

model. The endpoints are sorted on the basis of azimuth relative to the given (x, y) of the robot. The next step is to consider each line segment in the model and, making use of the sorted list of points, determine whether each point that lies in the azimuth range subtended by the current line segment either occludes or is hidden by the current line segment. The outcome of this step is that every corner in the model is labeled as either visible or invisible and, if visible, either occluding or nonoccluding. In one final pass, the array elements are stitched together into the ordered list of visible features required by the feature matcher. The worst-case complexity of the prediction algorithm is $O(N^2)$ (where N is the number of line segments in the model) in the case where there is a large number of occluded features. However, the average-case complexity is closer to $O(N \log N)$ for typical mine maps, especially if the feature predictor uses the maximum range of the scanner to limit the number of model lines considered.

Neither Drumheller [5] nor Krotkov [10] address this issue of feature prediction since they attempt to match all model features to all sensor features. Crowley [4] maintains a "composite local model," which consists primarily of the currently visible segments only.

E. Feature Extraction

Feature extraction is the process of reducing the raw range scan to an ordered list of features. The feature extraction algorithm first subdivides the scan into segments by breaking the scan at points where the distance between successive points exceeds a threshold, thereby finding occlusions. Pieces of the scan that contain only a small number of points (such as two or three) are thrown out since they give very unreliable features. Each piece of the scan is processed using a *corner operator* to find line segments and corners. The corner operator is a template consisting of two line segments that is compared against a sequence of range points to determine whether they exhibit a corner-like shape (see Fig. 5). The corner operator is parameterized by size (length of the component line segments) and by the angle between them. The corner operator template is "applied" to the range sequence by fitting the left and right line segments to the range data at a particular point using an incremental line fitter. If the template matches, a corner is defined at the intersection of the two line segments that form the operator. A larger corner operator gives more reliable feature parameters. Use of an incremental line fitter is possible because the corner operator template is moved sequentially one point at a time through the range scan and greatly decreases the amount of processing compared with a separate line fit at every point in the scan.

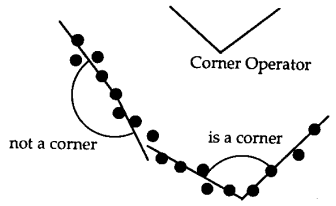


Fig. 5. Corner operator is applied at every point in the scan.

To find the parameters (length, perpendicular distance, orientation) of the line segment that lies between any two corners, a line is fit to all the data points in the scan sequence that fall between the two corners. The perpendicular distance and orientation are determined by the slope and intercept of the fit line; the length of the line segment is the distance between the two corners.

F. Feature Matching

The matcher determines the correspondence of the sensor features to the model features. A total of $N!$ sets of correspondences exist, assuming there are N model features and N sensor features. This prohibitively high complexity is further exacerbated by extra and missing sensor features corresponding to features not in the map (e.g., people, machines) and features that cannot be reliably extracted, respectively. Fortunately, the problem is usually constrained enough that only a much smaller set needs to be examined. See Wallace [12] for a survey of matching techniques. The type of matching technique used was determined by the following problem characteristics:

1) The position of the robot is approximately known.
2) The extracted features are reliable and rich in geometric information.

The first characteristic greatly constrains those sensor features that can match a model feature (called unary constraints), and the second greatly reduces the number of sensor-model pairs that need to be compared to determine match consistency. Three comparisons are used; hence, the constraint is labeled ternary. The algorithm

begins by applying the unary constraints to construct the set of possible sensor-model pairs. Then, the ternary constraints are applied to find the largest set of consistent correspondences. Efficient cut-off techniques are employed to reduce the complexity of the search.

The matcher first converts the list of predicted model features from the world coordinate frame to the robot coordinate frame using an approximate position given by dead reckoning; therefore, the model and sensor features are both in the robot coordinate frame. The unary constraints are then applied. These constraints are subdivided into *unary bounds* and *unary properties*. The unary bounds constrain the difference in position of corresponding sensed and model features based on the maximum position error (*poserr*) and orientation error (*angerr*) of the dead reckoning position. Specifically, the unary bounds can be stated as the following necessary conditions for correspondence between a sensor and model feature:

1) The angle to the sensed corner, relative to the positive x axis, must lie within $\pm$*angerr* of the angle to the model corner.
2) The distance from the origin to the sensed corner must lie within $\pm$*poserr* of the distance to the model corner.
3) The angle subtended by the sensed line segment must at least partially overlap (or be partially overlapped by) that of the model line segment, allowing a rotation of $\pm$*angerr* of the sensed line segment.

The unary property constraints are similar to the unary bounds constraints, except that whereas the unary bounds compare locations of features, the unary properties (for the most part) compare properties of the features that are unrelated to the position of the robot. Given that a particular candidate sensor-model pair satisfies the applicable unary bound constraints, the unary properties provide additional necessary conditions to further confirm the correspondence between a sensor feature and a model feature. The unary property constraints are specified as follows:

1) The concave angle of the sensed corner must lie within $\pm$*concerr* of the concave angle of the model corner.
2) The bisector orientation of the sensed corner must lie within $\pm$*angerr* of the bisector orientation of the model corner.
3) The orientation of the sensed line segment must lie within $\pm$*angerr* of the orientation of the model line segment.
4) The perpendicular distance to the sensed line segment must lie within $\pm$(*poserr* + *disterr*) of the perpendicular distance to the model line segment.

Concerr is the angular tolerance of the concave angle of a corner, and *disterr* is the distance tolerance of the perpendicular distance to a line segment. Each sensor feature is compared with every model feature, first with the unary bounds constraints and then with the unary properties constraints. In this way, the number of candidate sensor-model pairs (which before the unary constraints are applied is of size MN, where M is the number of model features, and N is the number of sensed features) is greatly reduced. The result is a list of ordered sensor-model pairs. The unary constraints, however, are not perfect, nor do they capture all of the available constraining information. Additional pruning is needed.

In order to extract from this ordered list of sensor-model pairs a set of consistent pairings, the next step utilizes the innate ordering (features are encountered in the order that the sensor scans) of the list and a set of ternary constraints to weed out the inconsistent pairings. We assume that the largest such consistent set of pairings is most likely to be a correct match and provides the most accurate position estimate. A ternary constraint compares a sensor-model pair to two sensor-model pairs that precede it in the ordered list. These two comparisons fall into one or two of the following categories:

1) If one of the sensor-model pairs is a pair of line segments and the other is a pair of corners, then the model line segment must be positioned relative to the model corner in the same way as the sensor line segment and corner.
2) If both sensor-model pairs are pairs of corners, then the distance between the two model corners must be equal to the distance between the two sensor corners $\pm$*terndisterr*.
3) If both sensor-model pairs are pairs of line segments, then the difference in orientation between the two model line segments must be equal to the difference in orientation between the two sensor line segments $\pm$*ternangerr*.

Terndisterr is a tolerance on distance error, and *ternangerr* is a tolerance on angular error. An efficient search algorithm applies these ternary constraints to the sensor-model pairs. The algorithm builds sequences of ternary-consistent sensor-model pairs using a depth-first search of the ordered sensor-model pair list. The longest sequences of consistent pairs leading into and out of each sensor model pair are tracked to enable the search algorithm to terminate fruitless branches efficiently. Thus, the search process assembles progressively larger sets of pairs until it exhausts all possible combinations. The largest set of pairs is used to calculate the position estimate.

G. Position Estimation

The position estimator finds a transformation that minimizes the total error between all transformed model features and their corresponding sensor features. Any combination of line segment feature pairs and corner feature pairs can be used in producing an estimate. The error measures are different for the two feature types—for line segments, there are two error measures (difference in orientation and difference in perpendicular distance from the origin), and for corners, there are also two error

measures (difference in x and difference in y). An iterative (Gauss–Newton) method is used to converge on a solution. The quantity to be minimized is $F + JD$, where F is a vector of the four error measures, J is the Jacobian of the four error measures differentiated with respect to the three transform parameters (x, y, θ), and D is the vector of differences between the transform parameters on successive iterations. For this algorithm to converge to a correct solution, it must be guaranteed that the initial orientation error is less than 90°, which is well within the accuracy of dead reckoning. The algorithm weights features by the number of range data points that were used to compute them. A better scheme that uses error of feature fit is planned for future work.

IV. Experimentation

The techniques developed in this work were carried beyond formulation and simulation into implementation to navigate a physical robot through a mine corridor mockup. In this section, we describe our mobile robot testbed, laser range sensor, computing environment, mockup and experimentation to set the context for results.

A. The Locomotion Emulator

The Locomotion Emulator (LE) is a mobile robot that was developed at the CMU Field Robotics Center as a testbed for development of mobile robotic systems (see Fig. 6). The LE consists of a *locomotor*, which is a mechanism capable of completely general locomotion on a flat surface, and an *emulator*, which is a software environment that enables this mechanism to mimic the characteristics of different vehicles. The locomotor is a powerful all-wheel steer, all-wheel drive base with a rotating payload platform. The steering motions of the LE's three-wheel modules are belt driven by one motor; its three drive motions are belt driven by another. Two shock-isolated enclosures, located between the wheel modules, house the on-board electronics and computing. Shore power is provided to the LE via a tether. The emulation software supports the four most common steering modules of wheeled robots—unicycle, Ackerman, skid, and articulated (see Fitzpatrick [7]). An operator or computer host can issue commands specific to any of these configurations, and the LE generates appropriate motions of its steer, drive, and payload platform to replicate the motions of the target vehicle. For the purpose of the experiments reported here, we have used the LE in omnidirectional (unicycle) mode.

B. Cyclone

The cyclone laser range sensor (Fig. 7) was also developed at the CMU Field Robotics Center to acquire fast, precise scans of range data over long distances (up to 50 m). The sensor consists of a pulsed gallium arsenide infrared laser transmitter/receiver pair that is aimed vertically upward. Like a lighthouse, a mirror in the tower part of the scanner rotates about the vertical axis and deflects the laser so that it shoots parallel to the ground, creating a 360° field of view. The tower rotation is belt driven by a motor at speeds from 0.5 to 5 Hz. The resolution of range measurements is 10 cm; the accuracy is ± 20 cm. The angular resolution depends on the resolution of the encoder that is used on the tower motor because the encoder pulses are used to trigger the laser. Currently, the scanner acquires 1000 range readings per 360° scan.

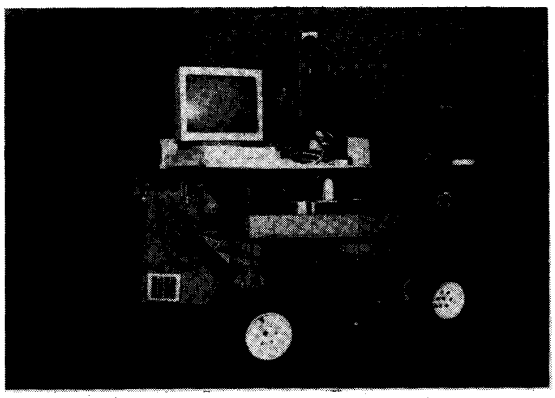


Fig. 6. Location emulator.

Fig. 7. Cyclone laser range sensor.

C. Computing Hardware

The perception system is implemented on a single processor (68030 20 MHz, 68882 math coprocessor) real-time development cage. A Sun 3 sits atop the LE but is used only to boot the real-time system and then to download files. A video board in the real-time cage displays relevant information to a color monitor.

D. Navigation Experimentation

All of our navigation experiments were conducted within a mine corridor mockup that covers an area of approxi-

mately 20 × 30 m of the highbay area of the Field Robotics Center. A photograph of this area appears in Fig. 8. The walls that are part of the line model include all of the partitions and two of the concrete walls of the highbay area (only one of them is visible in the photo—along the extreme left edge). Notice that the concrete walls are not perfect—there are a few roughly 0.4 m columns that jut out from the wall. Heavy clutter is often present (people, machines, equipment), none of which is represented in the line model. This clutter has never caused the perception system to fail in estimating a position; it only reduces the number of usable sensed features. We surveyed the mockup with a transit to create a line model. We generated intended paths as straight lines that direct the robot down the centers of the corridors.

The tracking algorithm takes one parameter (step size), which determines how far the robot moves between position estimates. After each estimate, the tracking algorithm moves to the point on the desired path that is a step size away from the current position estimate.

Fig. 9 shows the motion history of the robot along the desired path through the line model. Notice that there are two sets of arrows. The arrows that point exactly on path represent the desired positions, and the other arrows are the estimated positions. The discrepancy between the two sets of positions is due to a combination of dead reckoning inaccuracy (i.e., the robot did not go where it was told) and position estimate inaccuracy. Therefore, this plot does not isolate information about the accuracy of the position estimate.

Fig. 10 shows a raw range scan taken from the marked position with extracted features overlaid. The clusters of features at the top and on the right are due to clutter and are ignored by the matcher since there are no model features in these locations. This line model challenges the feature extractor due to the columns in the left wall, which have dimensions of about 0.4 m. To find the corners of these columns, a 0.4-m corner operator must be used, and since this dimension approaches the accuracy of the sensor (± 0.2 m), the corner operator is vulnerable to noise. Using a larger corner operator for this line model is not acceptable since it would miss these small features and treat the entire left wall as one line segment, columns included. The small corner operator produces some false corners, which are discarded by the matcher.

Figs. 11 and 12 compare two sets of features: Fig. 11 shows the predicted model features, and Fig. 12 shows the same set of extracted sensor features as in Fig. 11, except that they are transformed using the estimated position of the robot; therefore, it should ideally appear identical to the prediction. The numbers indicate the pairings of sensor and model features. In this case, there are 18 pairs of matched features. Our line model consists of 29 line segments, giving a total of 58 features. The feature predictor usually prunes the set to between 20 and 30, and the matcher usually matches between 10 and 20 features with some of the approximately 60 sensed features, requiring fewer than 250 recursive calls of the search function.

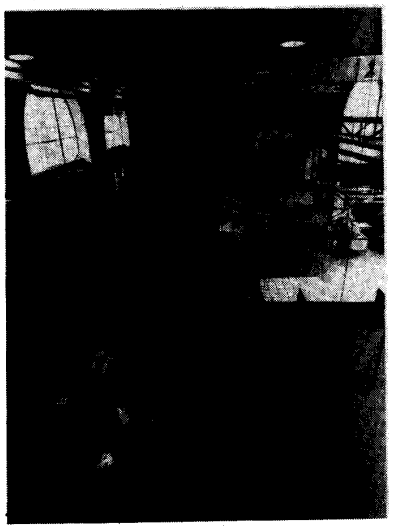


Fig. 8. Mine corridor mockup; corridor width is 6 m.

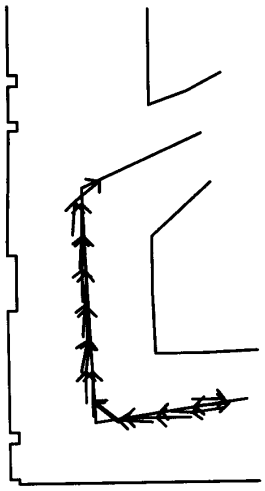


Fig. 9. Motion history of the LE in the middle of a run.

Thus, most position estimates are based on at least 10 feature pairs. Some of the model features match more than one sensor feature and vice versa; therefore, some numbers overlap in the figures.

E. Navigation Results

Fig. 13 shows an overlay of the path as reported by the position estimator, and the path as measured using the surveying transit. Table I shows the position data for the run of Fig. 13. The maximum position error is 0.40 m, and the average position error is 0.19 m. Table II shows the heading data for the same run. The maximum heading error was 1.9°, the average was 0.67°. The "correct" heading was provided by a gyro. The gyro drift problem was

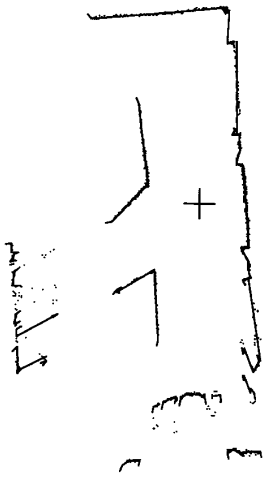


Fig. 10. Range scan with extracted features overlaid.

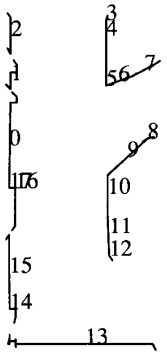


Fig. 11. Predicted features visible from current location of robot (marked with a cross in Fig. 10).

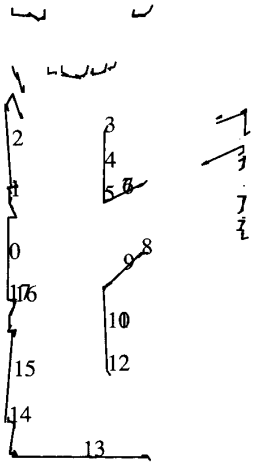


Fig. 12. Extracted features transformed according to position estimate. If the position estimate is perfect, this figure should almost be identical to Fig. 11.

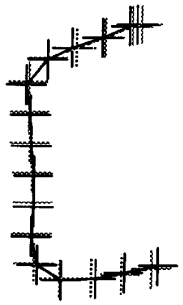


Fig. 13. Surveyed path (dotted) and estimated path (solid).

TABLE I
SURVEYED POSITIONS AND ESTIMATED POSITIONS

Surveyed x	Surveyed y	Estimated x	Estimated y
9.822m	3.646m	10.185m	3.804m
8.248m	3.353m	8.410m	3.442m
6.685m	3.158m	6.989m	3.134m
5.088m	3.099m	5.155m	3.033m
3.736m	3.921m	3.885m	3.824m
3.579m	5.483m	3.610m	5.320m
3.641m	7.082m	3.684m	6.869m
3.579m	8.694m	3.636m	8.520m
3.452m	10.261m	3.521m	10.112m
3.393m	11.854m	3.441m	11.697m
3.327m	13.425m	3.210m	13.292m
4.300m	14.660m	4.307m	14.614m
5.763m	15.258m	5.527m	15.236m
7.236m	15.793m	7.110m	15.802m
8.707m	16.392m	8.547m	16.416m
9.179m	16.517m	8.892m	16.489m

TABLE II
GYRO (ACTUAL) AND ESTIMATED HEADINGS

Gyro Heading	Estimated Heading
—	4.297°
187.587°	186.900°
181.628°	181.514°
177.675°	176.300°
143.068°	144.099°
89.324°	89.210°
81.704°	80.845°
87.090°	87.147°
88.179°	88.637°
87.090°	87.663°
86.689°	84.798°
45.035°	45.722°
15.241°	14.840°
12.834°	12.433°
18.163°	17.934°
10.714°	9.568°

virtually eliminated by using differences in gyro readings from position to position to update orientation, but this introduced some quantization error since the gyro reports heading to a resolution of 0.25°. Since the run consisted of 15 moves, the maximum error that this quantization produced was 1.875° by the end of the path. The parameter values used were *angerr* = 5°, *poserr* = 1 m, *concerr* = 10°, *disterr* = 0.5 m, *terndisterr* = 0.5 m, and *ternangerr* = 5.0°.

Approximate run times were 60 ms for feature prediction, 2 s for extraction, and 1 s for combined matching and estimation, giving a total cycle time of approximately 3 s.

F. Surveying Experimentation

In order to investigate the feasibility of using our laser range scanner and position estimation technique to survey an underground mine environment, we collected data from an operating underground coal mine in West Virginia. A sequence of scans was collected from accurately known positions in a 2-wk-old mine corridor. To collect data, we set up the Cyclone on a mobile mine cart and a surveying transit at a fixed reference point. Ten scans were recorded at each of 30 positions about 1 m apart. Each position was measured using the surveying transit. We had no way of measuring the orientation of the scanner. (Both pitch and roll of the scanner varied significantly as we moved.)

G. Surveying Results

Fig. 14 shows one of the scans with extracted features overlaid. The range data points in the middle of corridors correspond to either mine machinery (clutter) or hits on the roof. A large corner operator (3 m) was used since the geometry is large enough to allow it. The parts of the scene describable by line segments were successfully processed into line segments and corners.

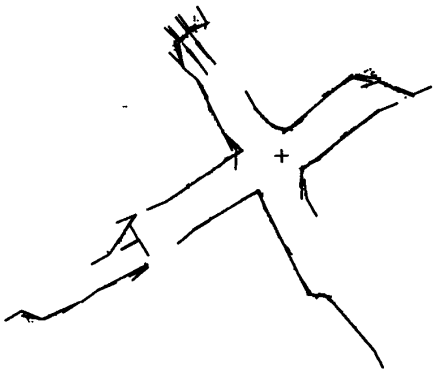


Fig. 14. Range scan from an underground coal mine with extracted features overlaid.

V. Conclusions

This paper presents a method of position estimation relevant to underground mine operations. It is a model-based perception technique that matches sensed features (from a laser range sensor) to predicted features (from a map). The world model is 2-D; features are line segments and corners. The resulting system has fast cycle time and high accuracy. It works well for environments that can be represented accurately by a 2-D line segment model.

Our experience suggests that the rounded corners and ribbed, wavy walls that occur in an actual mine do not appear in the idealized maps that are in use today. Although these nonideal features do not debilitate our system, they do degrade the position accuracy. Since some of the corners are not sharply defined in the mine, there is no one point that can be assigned as the vertex of that corner in order to form a line model, and by the same token, the feature extractor cannot lock on to one particular point either. In short, there is no guarantee of crisp corners. Although it is possible to add a third type of feature (the arc) to our system, it is clear that an iconic approach would have advantages for the mine environment. An iconic system would match all raw range points from a scan to a more detailed model of the world. The iconic system that we envision requires a more accurate initial estimate than dead reckoning can provide (especially for a continuous miner, which can hop around when cutting); therefore, the feature-based estimator would be used as a preprocessing step for the iconic system.

Future work will include implementation of the iconic position estimation system and a more sophisticated path planner and path tracker. We intend to first develop a more detailed world model that sufficiently represents the mine environment. A path planner that is based on this world model and incorporates the geometric and kinematic constraints of a continuous mining machine will be implemented long with a path tracker that will be capable of navigating a continuous miner along the planned path while avoiding obstacles.

Acknowledgment

The authors wish to thank B. Kumar for implementing the first version of the feature predictor and M. Blackwell and I. Lys for their contributions.

References

[1] B. C. Bloom, "Use of landmarks for mobile robot navigation," in *Proc. SPIE Conf. Intell. Robots Comput. Vision*, 1985, vol. 579.

[2] Z. L. Cao, J. J. Roning, and E. L. Hall, "Omnidirectional vision navigation integrating beacon recognition with positioning," in *Proc. SPIE Conf. Mobile Robots*, 1986, vol. 727.

[3] M. P. Case, "Single landmark navigation by mobile robots," in *Proc. SPIE Conf. Mobile Robots*, 1986, vol. 727.

[4] J. L. Crowley, "Navigation for an intelligent mobile robot," *IEEE J. Robotics Automat.*, vol. RA-1, no. 1, Mar. 1985.

[5] M. Drumheller, "Mobile robot localization using sonar," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-9, 1987.

[6] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE J. Robotics Automat.*, vol. RA-3, no. 3, June 1987.

[7] K. W. Fitzpatrick and J. L. Ladd, "Locomotion emulator: A testbed for navigation research," in *Proc. 1989 World Conf. Robotics Res.*, May 1989.

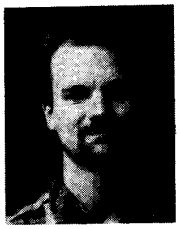
[8] G. Giralt, R. Chatila, and M. Vaisset, *An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots*. Cambridge, MA: MIT Press, 1984.

[9] A. Kak, K. Andress, and C. Lopez-Abadia, "Mobile robot self-location with the pseiki system," *Working notes AAAI Spring Symp. Robot Navigation*, Mar. 1989.

[10] E. Krotkov, "Mobile robot localization using a single image," in *Proc. IEEE Robotics Automat. Conf.*, May 1989.

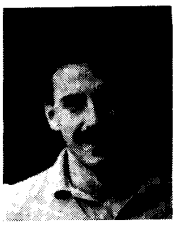
[11] H. P. Moravec and A. Elfes, "High resolution maps from wide-angle sonar," in *IEEE Int. Conf. Robotics Automat.*, Mar. 1985.

[12] A. M. Wallace, "A comparison of approaches to high-level image interpretation," *Patt. Recogn.*, vol. 21, no. 3, 1988.



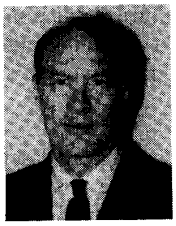
Gary K. Shaffer received the B.S. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 1989.

He is a Ph.D. student in the Robotics Institute at Carnegie Mellon. His research interests include mobile robot planning, perception, and control. Current research is focusing on building 2-D maps from range data. He has been part of several research efforts including automation of a strip mine dump truck, automation of a manipulator for nuclear inspection tasks, and on-going research in automation of underground coal mining.



Anthony Stentz received the Ph.D. degree in computer science from Carnegie-Mellon University in 1989. He received the M.S. degree in computer science from Carnegie Mellon in 1984 and the B.S. degree in physics from Xavier University of Ohio in 1982.

He is a research scientist in the Robotics Institute at Carnegie Mellon University. His research interests include mobile robots, path planning, computer vision, system architecture, and artificial intelligence in the context of fieldworthy robotic systems operating in unstructured environments. He has led a number of successful research efforts to automate navigation and coal cutting operations for a continuous mining machine, cross-country navigation for an unmanned ground vehicle, and nuclear inspection tasks for a manipulator arm. His new activities include automating the takeoff, landing, and navigation operations for the Unmanned Air Vehicle and digging operations for a robotic excavator.



William L. Whittaker received the B.S. degree from Princeton University in 1973 and the M.S. and Ph.D. degrees from Carnegie Mellon in 1975 and 1979, respectively. He is currently a Principal Research Scientist with the Robotics Institute at Carnegie Mellon University, Pittsburgh, PA.

His research interests center on mobile robots in field environments such as work sites and natural terrain and include computer architectures to control mobile robots, modeling, and planning for nonrepetitive tasks, complex problems of objective sensing in random or dynamic environments, and integration of complete field robot systems.

Projects under his direction include an Unmanned Ambler to explore planetary surfaces, automation of mining machines, remote working machines for hazardous waste cleanup in the nuclear industry, and autonomous land vehicle navigation.

Dr. Whittaker is the recipient of numerous awards, including Carnegie Mellon's Teare Award for Teaching Excellence. Science Digest named him one of the top 100 U.S. innovators in 1985 for his work in robotics. He is also cofounder and chief scientist of Redzone Robotics.

Kerien W. Fitzpatrick received the B.S. degree in civil engineering from Carnegie Mellon University in 1985.

His research interests center on mobile robots operating in unpredictable and unstructured environments. While working as an undergraduate, he participated on projects such as the Remote Reconnaissance Vehicle and the Terregator. After graduation, he was hired as full-time staff and has participated in and supervised projects such as the Remote Work Vehicle, the Locomotion Emulator, the HMMWV, and the Unmanned Air Vehicle. The range of projects in which he has been involved has provided him significant experience in the conception, design, and fabrication of mobile robots. In addition to his roles as a mechanical design and project manager, he is also responsible for all aspects surrounding the computing facilities for his current employer.