An Autonomous Excavator with Vision-Based Track Slippage Control

Parvaneh Saeedi, Peter D. Lawrence, David G. Lowe, Poul Jacobsen, Dejan Kusalovic, Kevin Ardron, and Paul H. Sorensen

***Abstract*— This paper describes a vision-based control system for a tracked mobile robot (an excavator). The system includes several controllers that collaborate to move the mobile vehicle from a starting position to a goal position. First, the path planner designs an optimum path using a predefined elevation map of the work space. Second, a fuzzy logic path-tracking controller estimates the rotational and translational velocities for the vehicle to move along the pre-designed path. Third, a cross-coupling controller corrects the possible orientation error that may occur when moving along the path. A motor controller then converts the track velocities to the corresponding rotational wheel velocities. Fourth, a vision-based motion tracking system is implemented to find the 3D motion of the vehicle as it moves in the work space. Finally, a specially-designed slippage controller detects slippage by comparing the motion through reading of flowmeters and the vision system. If slippage has occurred, the remaining path is corrected within the path tracking controller to stop at the goal position. Experiments are conducted to test and verify the presented control system. An analysis of the results shows that improvement is achieved in both path-tracking accuracy and slippage control problems.**

***Index Terms*— Robot vision, visual motion control, trajectory control, path planner, track slippage, vehicle position sensing, excavator control, motor control, vision-based trajectory estimation.**

I. Introduction

TRACKED vehicles, such as excavator-type machines, are widely used in industries such as forestry, construction and mining. These machines are used for a variety of tasks, such as lifting and carrying loads, digging and ground leveling. Autonomous controls for driving or assisting humans in operating these machines can potentially improve the operational safety and efficiency. Much research has gone toward controlling vehicle movement to reduce human interaction when the vehicle is performing a task [1] [2]. Removing the operator for direct control of the machine has been achieved in teleoperation [3] [4] [5]. Achieving this goal in natural environments requires planning every movement, to avoid any obstacles and to locate the vehicle at each time with respect to a global coordinate system. With the application of an effective control scheme, human error can be minimized or completely removed, and more consistent operation of the vehicle can be achieved to increase efficiency.

Many methods for outdoor path planning have been developed to find feasible paths from start to end locations [6] [7] [8]. Some of these methods rely on a prior map of the environment, to be provided by an operator. Since a static map is not sufficient, due to outdoor changes, other methods use sensory devices such as laser range finders and stereo cameras to create and update their own neighborhood maps as the vehicle navigates.

Numerous methods have also been developed to track trajectories and paths outdoors [9]–[15]. Some of these methods implement non-linear trajectory control algorithms using the difference between the actual and virtual reference positions. Others accomplish the task by generating error vectors from the lateral displacement and heading errors. Path tracking has also been performed using feedforward compensation for the steering mechanism to provide anticipatory control of the steering lag.

Many sensors have been developed and applied to robot localization. Wheel sensors such as encoders and resolvers suffer from wheel-slip. Estimating position using angular rate sensors and accelerometers suffers from integration error due to the effects of noise. Sonar, a low resolution system, is sensitive to environmental disturbances (wind, temperature, foliage motion and machine noises), Laser range scanning is expensive and has a low image update rate. Even GPS, can suffer from occlusion of line-of-sight to satellites, low accuracy, and low update rate. Since solid-state cameras and computers are rapidly improving in quality and speed driven by a consumer market, they are an tempting platform on which to build a robot localization system.

The main long-term goal of this work is to move a tracked vehicle from a starting point to a target point in an unstructured outdoor environment. This paper describes the design and implementation of a complete system for a Takeuchi TB035 excavator that contains all of the subsystems necessary for this task. In particular, path planning, path tracking control, excavator track control, and visual motion tracking were selected as necessary building blocks. To make the overall development feasible in a reasonable time and to allow for real-time performance, some constraints were applied. The system was constrained to plan and execute straight line segments and small-radius turns, and that the vision system would be angled down to avoid sun, and thus be protected from raindrop accumulation - limiting the scene to a local region only. A unique aspect of this work, enabled by the vision system, is to detect and correct track slippage. Excessive track slippage can take place on slopes in various ground conditions, and can lead to soil erosion. The proposed path planner can be tuned to find paths that will reduce the possibilities of excessive slippage.

II. Motion Control System Overview

A block diagram of the semi-autonomous vehicle motion control system is shown in Figure 1. The human operator

stays at the top level of the entire control system. The operator chooses between manual or autonomous control of the vehicle. If manual control is desired, the operator can drive the vehicle and the autonomous controller becomes disabled. If autonomous control is chosen, the operator selects a desired goal posture $P_g = (x, y, \phi, V)$; here x and y represent the location and ϕ and V show the orientation and the path speed for the vehicle. This position, along with a map of the environment, is passed to a path planner unit. The path-planner designs a collision free optimal path from the current posture of the vehicle, P_c, to the desired goal posture, P_g [16]. The optimal path is generated based on several cost functions, derived from the slope steepness and type of terrain. These cost functions try to minimize the environmental impact of the vehicle.

Once the path is chosen, the path tracking controller is responsible for keeping the vehicle moving along that path. Due to the nonlinearities of the tracked excavator and the existing uncertainties in a natural environment, a fuzzy logic scheme was used for the tracking controller. In using this approach not only are the control rules expressed easily in linguistic statements but also the allow a more flexible design with respect to the existing uncertainties in natural environments. At each time, the current path segment is passed to the path tracking controller. By comparing the current position of the vehicle (obtained incrementally from the flowrate sensors) with the path to be followed, the reference translational and rotational velocities for the mobile robot are computed. The vehicle is either commanded to continue driving along the path, when on the desired path, or made to converge back to the desired path if it has strayed. When the vehicle reaches the end of a current requested segment, the path tracking controller requests the next path segment from the path planner. The corresponding left and right track reference speeds, $V_{D,L}$ and $V_{D,R}$, are then obtained by the application of the inverse kinematics equations of motion. The reference speeds can be adjusted using a coefficient factor, α, determined based on the occurrence of excessive slippage.

The cross-coupling motion controller (CCMC) controls the heading error [17] directly, using the left and right track reference speeds, as well as the accumulated distance error between the two tracks. The distance error is estimated by subtracting the distances traveled by the left and right tracks, obtained by integrating the flowmeters readings. The output of the CCMC are the individual track speeds, ($V^{'}_{R,L}$, $V^{'}_{R,R}$), used in the track controller. The cross-coupling controller has been used for the excavator since it corrects for orientation error by integrating flowmeter information for left and right tracks.

The track control and motor subsystem block was experimentally modeled as a first-order transfer function with delay. The two track control and motor blocks operate independently of each other to produce the rotational velocities of the track ω_L and ω_R.

The sensors on the vehicle, rate flowmeters in the hydraulic lines provide the track motor rotational velocities used for dead-reckoning position estimation. The readings of these sensors along with the odometery equations are used to update the current location of the mobile robot.

The vision-based tracking system includes a camera head with three on-board CCD cameras, which can be mounted anywhere on the vehicle. The system processes consecutive trinocular stereo sets of images to detect and track correspondences to the most stable points in the environment. Using stereo cameras and measuring the $2D$ displacements of similar image features in different frames, the $3D$ trajectory of the camera, and hence, the vehicle is estimated. The integration of an individual scene-independent vision-based tracking system delivers an independent control, at a higher level, for identification and correction of the track slippage problem.

If track motion calculated by the dead-reckoning sensors sufficiently exceeds the track motion calculated by the vision system, for the same time period, a slippage value is calculated for the slippage controller. The slippage controller uses the slippage value to compute a scaling factor, α, to reduce the left or right track reference speeds, $V_{D,L}$ and $V_{D,R}$. Once the slippage is eliminated, the scaling factor is reset to unity and the reduction in the track reference speeds is removed.

Each one of the introduced components is presented in more detail in the following sections. The paper is organized as follows. The path planner is briefly explained in Section III. The design and implementation of the path tracking controller is described in Section IV. The cross-coupling motion controller is studied in Section V. The motor controller is addressed in Section VI. Section VII represents the vision based motion tracking system. The slippage controller is discussed in Section VIII. Experimental results are represented in Section IX. Conclusions and future work are outlined in Section X.

III. Path Planning Controller

Autonomous navigation in outdoor environments requires a path planning system that can identify a feasible path quickly and effectively. This section describes path planning system that finds an efficient and safe path for autonomous navigation in small or medium size outdoor environments.

There are two main objectives associated with the design of the path planner. First, it must find an obstacle free path, from a starting point to a goal point. Second, it has to minimize soil disturbance, in forestry applications for example, while operating heavy machinery outdoors. The inputs to the system are the elevation map of the work space, as well as cost constraints. The obstacles are defined as either objects that block a specific path, or as terrain that is not traversable due to mechanical constraints of the machine (for example steep slopes). In terms of efficiency, the shortest path length is considered here to be the main criterion.

Figure 2 represents the flowchart of the implemented path planner. First, the resolution of the environment grid and the type of cost functions are determined in the set up process. Next, the goal position is decided by the operator. Then, using the input map and cost functions, a cost map is created. A search routine then finds the optimized solution path. Finally, special cases where a feasible solution cannot be found are

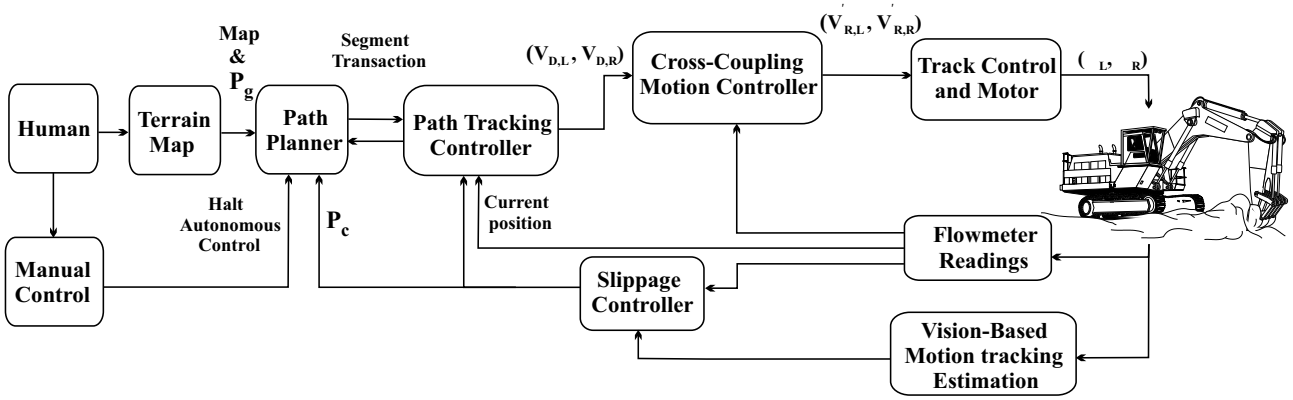


Fig. 1. Motion control system block diagram.

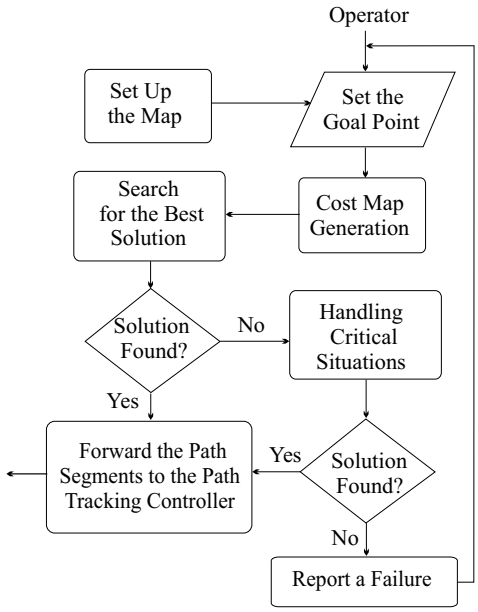


Fig. 2. The flowchart of the path planning algorithm.

re-investigated. The details of each block are explained later in this section.

A. User's Set Up

The first user's setup parameter is the resolution of the map grid, $n \times m$, which shows the number of existing nodes across the entire workspace. The second input to the system is the elevation map of the work space, produced either from satellite images or airborne sensors and transformed into a $2D$ graph using the area's grid map. Stereo camera-based images from aircraft are commonly used to create surface elevation maps. Radarsat and aerial laser range finders have also been used to create elevation maps. Currently digital elevation maps of the 7,027 map sheets, covering the province of British Columbia, Canada, at a scale of 1:20 000 are produced by the TRIM program (Terrain Resource Information Management Program) [18]. The digital mesh that is used in TRIM has a 20 meter interval spacing with an accuracy in spacing of 10 meters and in altitude of 5 meters. Since a denser map would be more desirable and would result in a more precise path planning, through all of the presented experiments, the elevation map is created in a finer grid. It is expected that commercial resolution will improve over time.

B. Cost Map Generation

As explained earlier, one of the requirements of the path planner is to optimize the found path, based on several conditions such as length, slope and soil disturbance. However, employing a multiple objective search can significantly impact the required memory and increase running time dramatically. Also, some of the constraints in the cost functions may be mutually conflicting. For these reasons, for every existing path a cost value is associated with a weighted sum of individual constraints [19]. A maximum number of six weight coefficient combinations is allowed in the design. This number permits a wide range of choices likely to be suitable in different situations.

The cost function associated with every traversable arc in the map is a scalar calculated from:

$$c(i,j) = \lambda_1 . Arc\ length + \lambda_2 . \frac{Elevation\ change}{Arc\ length} \quad (1)$$

$$Arc\ length = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \quad (2)$$

$$Elevation\ change = |z_i - z_j| \quad (3)$$

In Equation 1, $Arclength$ reflects the objective of traveling on the shortest possible path. The term $Elevation\ change/Arc\ length$ allows the algorithm to find a path that will have a small elevation change (by going around a hill, for example, to avoid an elevation change) but have a longer path length. In case it is necessary to climb a slope, due to the fact that the second term is optimizing the ratio of the elevation to the length, the cost function will favor the path that reaches the goal by climbing a hill in a spiral fashion. If the slope is not a rounded hill but a width-limited inclined slope, the path turns out to be a zigzag path. The local angles available are 0^o, 45^o, and 90^o due to

the fact that the local movement of the vehicle is limited to the neighboring nodes in the graph. The influence of the two terms of the cost function is balanced by two parameters λ_1 and λ_2. The relationship between coefficients λ_1 and λ_2 can be expressed by:

$$0 \leq \lambda_1 \ , \ \lambda_2 \leq 1 \tag{4}$$

$$\lambda_1 + \lambda_2 = 1 \tag{5}$$

A value of $\lambda_1 = 1$, for instance, is the solution for the classical shortest path optimization. There are a maximum of six different combinations for λ_1 and λ_2 which are predefined. These combinations are shown in Table I.

TABLE I

PREDEFINED VALUE COMBINATIONS FOR (λ_1, λ_2).

N	(λ_1, λ_2)
1	(1,0)
2	(1,0), (0,1)
3	(1,0), (0.5,0.5), (0,1)
4	(1,0), (2/3,1/3), (1/3,2/3), (0,1)
5	(1,0), (3/4,1/4), (0.5,0.5), (1/4,3/4), (0,1)
6	(1,0), (4/5,1/5), (3/5,2/5), (2/5,3/5), (1/5,4/5), (0,1)

Here, N determines how many searches will be performed. The purpose of having different choices for this combination is to reduce the computation time in cases where suitable weight combinations can be predicted given the conditions. The goal is to see whether there is a possible path among six candidates. In general, it is more likely that a superior path will be found for larger values of N. In this implementation however N is limited to 6, which represents an engineering trade off between the number of choices and the execution time.

C. Search for the Best Solution

The accuracy and efficiency of the path planning controller is mainly dependent on the search method's convergence and computational complexity. Among the numerous existing search algorithms, the A* approach was chosen for its robustness in off-road route planning [20] [21]. This method has a simple yet robust structure that allows specific constraints to be incorporated easily [22] [23].

The algorithm begins at a start node and then iteratively checks surrounding nodes for their accessibility. When a traversable node is visited for the first time, a backpointer to the previous node, as well as the cost of the path traveled so far, is estimated and assigned to the node. Each time a traversable node is accessed from one of its neighbors, the new path cost is compared with the previous best; if it is less, the backpointer is redirected, Figure 3.The cost function associated with a path to node n is expressed by:

$$f(n) = g(n) + h(n) \tag{6}$$

where $g(n)$ is the sum of costs associated with the arc leading from the start to the node n. Function $h(n)$ is a heuristic estimate of the cost from node n to the goal node. The A* algorithm guarantees finding the cheapest solution path if one exists; otherwise it reports a failure. The original A* approach is a single objective. However, the system reported here requires the discovery of an optimized path with multiple objectives. This problem is overcome by introducing one cost function for each objective, and combining the cost functions into a single objective function.

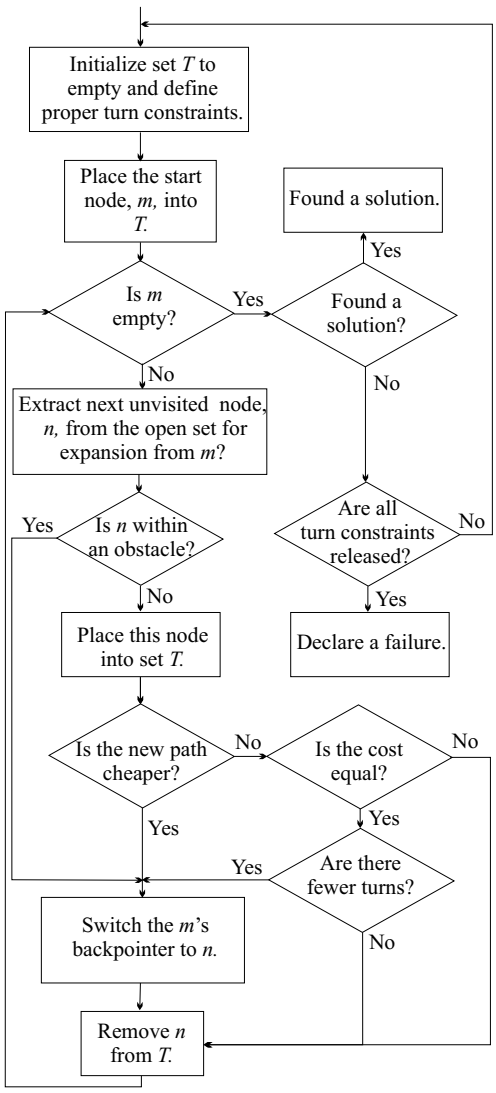


Fig. 3. The search algorithm block diagram.

The grid map of the workspace confines the angle changes with respect to the previous arc to 0^o, 45^o, 90^o, 135^o and 180^o. Sharp angle changes, for instance 135^o, can cause undesirable disruption of delicate forest soil. To avoid this, two strategies are considered.

First, rotation angles $\geq 90^o$) are banned. This requirement has the highest priority. For this purpose the search process is performed in three consecutive steps. On the first try, the angles of 135^o and 90^o are banned. Whenever a new optimal path is discovered, it is checked for any banned angles with the two node's predecessors. If any banned angle exists, then an obstacle is reported and the backpointer is not switched. This constraint, however, reduces the connectivity of the graph and may spoil the A* indigenous convergence. For instance, it is possible to declare a failure though a path actually exists. This happens where there is a narrow (one node wide) channel

containing banned angles on every possible path. If the first search attempt fails, the constraint set bans turns of 135^o only. If a path is again not found, a search is performed with no constraints. In this way, when a solution is found then the algorithm convergence remains unaffected.

The second strategy minimizes the number of turns in general. This is enforced by employing one more test when a new path is discovered. In the original A* algorithm, if the cost of a new path is the same as the previous best, the backpointer is not switched, even if one of these paths contains more direction changes. This cost is modified by comparing the number of turns and switching to a neighboring node with fewer turns. This feature does not effect convergence, since it does not involve recalculation of the cost function. If the paths (backpointer nodes) are switched, the procedure continues as usual.

D. Handling Critical Situations

One of the safety features included in the system is implemented by expanding the edges of found obstacles through grid node displacement. This safety feature is paid for by the possibility of not finding a solution even when one may exist. To overcome any system failure due to obstacle border expansion, a second search is conducted after the obstacles' borders shrink back to their original positions. If the system still cannot find a solution, then a failure is announced.

E. Path Planner Evaluation

To evaluate the functionality and efficiency of the implemented path planner, several different terrain types are tested. The test subjects include flat terrains, inclined planes, and narrow hills with significant or severe slopes. Some of the results are presented here. Figure 4 represents the implementation results for slopes.

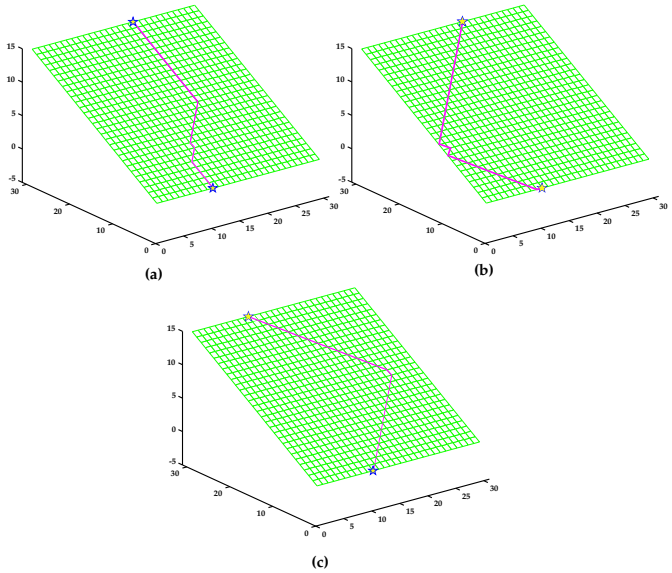


Fig. 4. Path planning outputs for wide slopes.

Figures 4-a and 4-b show paths designed for traveling between two points. The constraints for these two paths are designed as (λ_1=0.8,λ_2=0.2) and (λ_1=0,λ_2=1). These two conditions do not, by themselves, satisfy the system requirements. This is because the first solution is neither satisfactory regarding the soil disturbance, nor safe, due to rapid elevation change. The second solution includes more safety considerations, but it is long and soil disturbance remains a problem. Figure 4-c shows an acceptable path. Since the traversable area for this path is wide, there are only two turns.

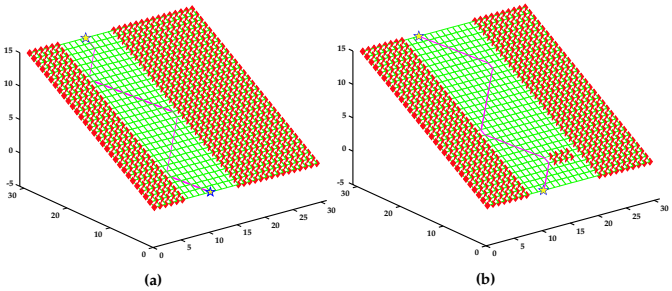


Fig. 5. Path planning outputs for narrow slopes with and without obstacles.

Figure 5 shows the output for a narrow slope with and without obstacles. The path planner follows the same principles of not climbing directly upward, and to minimize the number of turns. At the beginning of its route, it makes more turns (Figure 5-b), since the path arcs are narrower when the obstacle occupies some of the free space. The path planner gradually increases the path arcs and reduces the number of turns as the obstacle is passed and the unoccupied area becomes wider. Figure 6 represents a set of solution paths

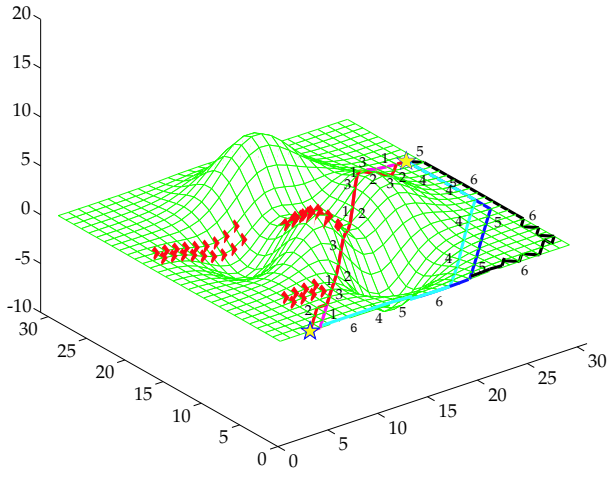


Fig. 6. Path planning outputs for a hilly surface with obstacles.

for a severely hilly configuration space with three obstacles, found by the system using different combinations of λ_1 and λ_2 (defined in row 6 Table I). Table II defines the corresponding λ_1 and λ_2 for each path based on its number on Figure 6.

TABLE II
COMBINATIONS OF (λ_1, λ_2) FOR PATHS FOUND IN FIGURE 6.

Path number	(λ_1, λ_2)
1	(1,0)
2	(0.6, 0.4)
3	(0.4, 0.6)
4	(0.8, 0.2)
5	(0.2, 0.8)
6	(0,1)

Path 1 has the shortest length while path 6 represents the solution with minimum elevation change. From these results, it can be seen that in paths 2 to 5 fewer turns have been taken while still avoiding slopes when possible. On the contrary, solution 6 can create unnecessary long paths because with a flat area all arcs have the same cost (0).

In summary, the path planner generates multiple solution paths using different weight combinations in the cost function. For each path, 6 different solutions are created and one is either selected based on total cost and number of turns, or all can be offered to an operator to choose from. The wide range of existing factors in an industrial or outdoor work space (soil type, saturation of a terrain, etc.) make the approach suitable for such environments. It is interesting to note that some parts of the algorithm need to be performed only once for all 6 paths and that the software design is easy to implement on a parallel CPU which would cut the execution time to very close to that of the single choice algorithm.

After a path solution is found, it gets segmented and passed to the path tracking controller upon its request. In the next section, the principles of the path-tracking controller is discussed.

IV. PATH TRACKING CONTROLLER

Figure 7 represents the block diagram of the path tracking controller. The duty of this controller is to maintain the mobile vehicle on a desired path.

The inputs to this controller are the current position and orientation of the vehicle, and the information relating to the desired path segments. Using these two pieces of information, the controller estimates the motion needed to reach the end of the desired path segment. Upon reaching the end of the desired path segment, the path tracking controller transmits a path segment request to the path planner for the next path segment to be traversed.

The outputs of this controller are the desired left and right track velocities. Details regarding the role of each component in this controller, and its relationship with other controllers are explained in the following sections.

A. Trajectory Error Generation

The trajectory error generation block generates an error vector, $(x_{v,e}, y_{v,e}, \phi_{v,e})$, in the vehicle frame using the desired destination, $(x_{d,w}, y_{d,w}, \phi_{d,w})$, and the current position, $(x_{c,w}, y_{c,w}, \phi_{c,w})$, (Figure 8). The desired destination is provided by the path planner, while the current trajectory is estimated through the readings of the flowmeters. The gener-

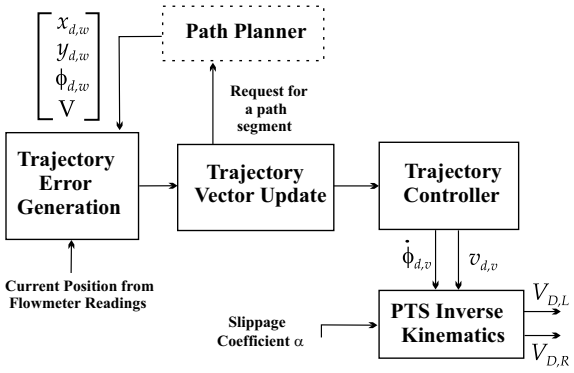


Fig. 7. Block diagram of the Path Tracking Controller.

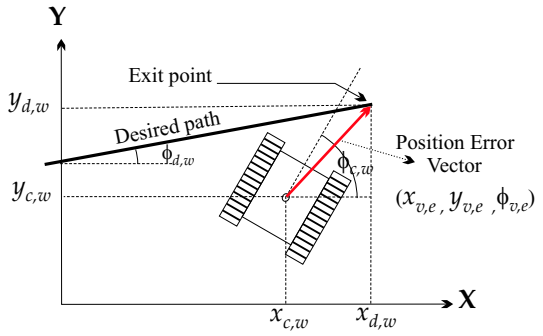


Fig. 8. Trajectory error generation in the path tracking controller.

ated trajectory error, output of the Trajectory Error Generation block in Figure 8, can be expressed by:

$$\begin{bmatrix} x_{v,e} \\ y_{v,e} \\ \phi_{v,e} \end{bmatrix} = \begin{bmatrix} cos(\phi_{d,w}) & sin(\phi_{d,w}) & 0 \\ sin(\phi_{d,w}) & -cos(\phi_{d,w}) & 0 \\ 0 & 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} x_{d,w} - x_{c,w} \\ y_{d,w} - y_{c,w} \\ \phi_{d,w} - \phi_{c,w} \end{bmatrix} \quad (7)$$

Equation 7 represents the position error vector as the rotated difference of the x and y coordinates of the current and desired locations by the desired orientation. This error represents the positional correction required before reaching to the end of the current path segment.

B. Trajectory Vector Update

The trajectory vector update, Figure 7, is responsible for requesting a new segment from the path planner when the end of the current segment is reached. This is achieved when the value of $x_{v,e}$ approaches zero. The trajectory vector update also derives the error changes of $\frac{\Delta x_{v,e}}{T_s}$, $\frac{\Delta y_{v,e}}{T_s}$ and $\frac{\Delta \phi_{v,e}}{T_s}$ that are used in the trajectory controller where T_S is the sampling interval.

C. Trajectory Controller

The translational and rotational speeds of the vehicle, $(v_{d,v}, \dot{\phi}_{d,v})$, are controlled by this controller. The translational control is performed by determining the desired translational speed of the vehicle from the changes in $x_{v,e}$ and $y_{v,e}$. The orientation control is performed by using $y_{v,e}$ and $\phi_{v,e}$ and

their derivatives to derive the desired rotational velocity of the vehicle. The goal is to converge the inputs to zero:

$$y_{v,e} = 0\ , \quad \frac{\Delta y_{v,e}}{T_s} = 0 \tag{8}$$

$$\phi_{v,e} = 0\ , \quad \frac{\Delta \phi_{v,e}}{T_s} = 0 \tag{9}$$

1) Translational Speed Controller: Regardless of the position and orientation, this controller is responsible for maintaining a desired acceleration during the starting time and speed change when converging to a desired translational speed. It is also responsible for stopping at an exact, user specified position with a desired deceleration. Here, the sum of changes in x and y of the vehicle are measured to estimate the actual speed of the vehicle. The control rules for converging to, or maintaining, a desired action can be expressed as:

$$\frac{dv_{d,v}}{T_s} = \begin{cases} -A_v, & \text{if } v > V \\ A_v, & \text{if } v < V \end{cases} \tag{10}$$

$$\text{where } v = \sqrt{\left(\frac{\Delta x_{v,e}}{T_s}\right)^2 + \left(\frac{\Delta y_{v,e}}{T_s}\right)^2} \tag{11}$$

These equations state that the desired velocity of the excavator is increased (decreased) if the excavator Euclidean velocity V required to eliminate the error in one time step is less than (greater than) the velocity v specified by the path planner (where V is bounded by the maximum permitted excavator speed).

At each turning point the translational speed has to be decreased to prevent the vehicle from overshooting when converging to a new path. In order to ensure a stop at an exact stop position with a specific deceleration A_{Stop}, the following rules [24] are adopted:

$$v_{d,s}(x_{v,e}) = \text{sign}(x_{v,e}).\sqrt{2.A_{Stop}.|x_{v,e}|} \tag{12}$$

$$\frac{\Delta v_{d,v}}{T_s} = \begin{cases} 0, & \text{if } v_{d,s}(x_{v,e}) = V \\ A_{Stop}, & \text{if } v < v_{d,s}(x_{v,e}) \\ -A_{Stop}, & \text{if } v > v_{d,s}(x_{v,e}) \end{cases} \tag{13}$$

The performance of the translational controller was tested on a 1/14.2 scale mechanical tracked excavator [25] originally used to develop the trajectory controller. More tests are reported in Section IX in conjunction with the final full scale system. Figure 9 shows the results of one of these experiments. This experiment evaluates the controller's ability to accelerate smoothly from a start position, and to stop the vehicle at an exact position with a certain deceleration on a $100cm$ long path. Figure 9-a denotes the traversed path and Figure 10-b depicts the generated translational velocity by the designed controller.

2) Rotational Speed Controller: This controller forces the vehicle to move on a desired path by controlling the rotational speed, $\dot{\phi}_{d,v}$. For this purpose, two fuzzy PD-regulators, one for $y_{v,e}$ and its derivative and another for $\phi_{v,e}$ and its derivative, are implemented.

The input space of the fuzzy-PD controllers are $y_{v,e}$, $\frac{dy_{v,e}}{T_s}$, $\phi_{v,e}$ and $\frac{d\phi_{v,e}}{T_s}$. Variables y and ϕ are divided into five

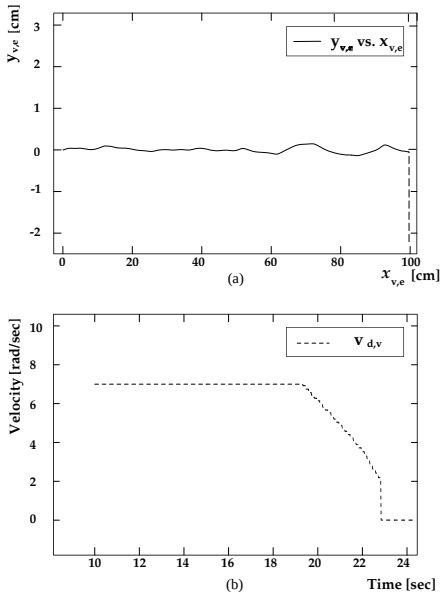


Fig. 9. The output of the translational velocity controller.

membership functions (MFs) and $\frac{dy_{v,e}}{T_s}$ and $\frac{d\phi_{v,e}}{T_s}$ into three. Figure 10 describes the partitions and the naming conventions for y and ϕ. The span of each MF is described by a triangular shape. For variable ϕ the overall span is extended between $-\pi$ and π, (Figure 11-a). The span around zero, $Ephizero$, is defined to be very narrow, -15^o to 15^o, since even a small angle error is not acceptable when attempting to navigate on an exact path. $Ephismp$ and $Ephismn$, small positive and small negative errors, MFs are designed for situations in which the vehicle is close to the right direction and consequently, the velocity has to be restrained to avoid an overshoot occurrence. The ranges of these two groups are defined in $[0^o\ 35^o]$ and $[0^o\ -35^o]$. Finally, the medium positive and negative MFs, $Ephimep$ and $Ephimen$, describe the rest of the input by $[18^o\ 180^o]$ and $[-18^o\ -180^o]$.

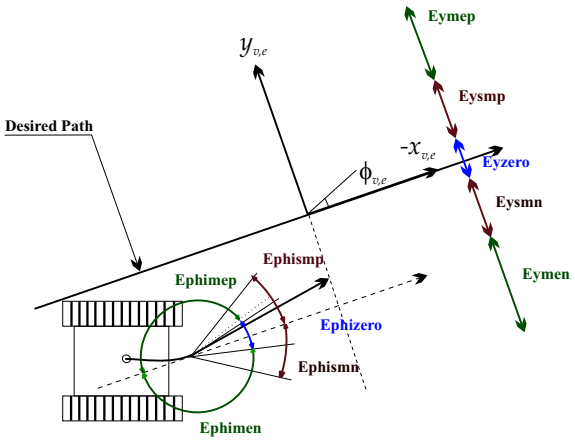


Fig. 10. Fuzzy logic rotational velocity controller input space.

Similarly the universe of y, dy/T_s and $d\phi/T_s$ are partitioned. Figures 11-b to 11-d show the span and range of the input variables. It can be seen that partitions of dy/T_s and

$d\phi/T_s$ are designed to describe the directions in which the y and ϕ move.

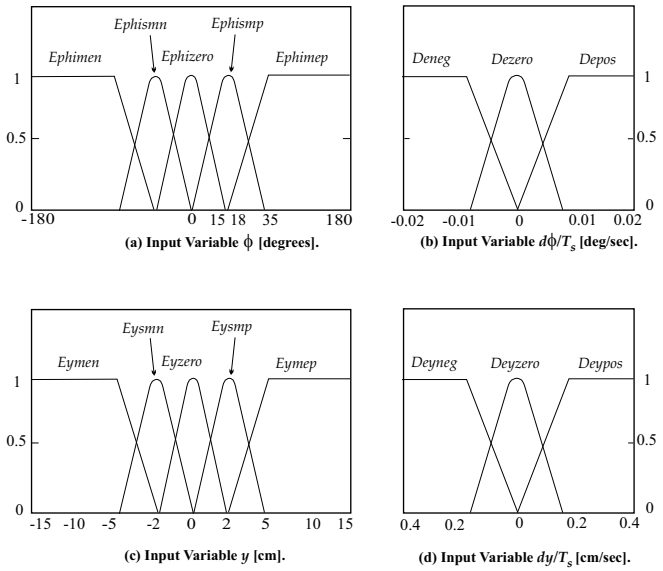
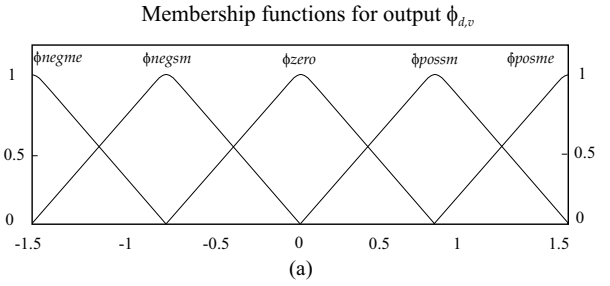


Fig. 11. Fuzzy logic input space membership functions.

The output space for this controller is defined by variable $\dot{\phi}_{d,v}$ with a defuzzication of the centroid-based method [26]. In order to have smooth output changes, the membership functions are equally distributed as shown in Figure 12-a. The output space mapping function rules that describe the behavior of the system under different input conditions, are represented in Figure 12-b. This Figure describes the traditional PD-controller responses in linguistic terms. Since the behaviors of the two rule sets (one with inputs ($\phi_{v,e}$, $d\phi_{v,e}/T_s$) and the other with inputs ($y_{v,e}$, $dy_{v,e}/T_s$)) governing $\dot{\phi}_{d,v}$ are very similar, they are summarized in one description table.



Input: $\phi_{v,e}$ / Input: $\frac{d\phi_{v,e}}{T_s}$ / $\frac{dy_{v,e}}{T_s}$	Ephimen / Eymen	Ephismn / Eysmn	Ephizero / Eyzero	Ephismp / Eysmp	Ephimep / Eymep
Dphineg / Dyneg	ϕposme	ϕpossm	ϕpossm	ϕpossm	ϕnegme
Dphizero / Dyzero		ϕpossm	ϕzero	ϕnegsm	
Dphipos / Dypos		ϕnegsm	ϕnegsm	ϕnegsm	

(b)

Fig. 12. Fuzzy logic rotational velocity control output space.

D. PTS Inverse Kinematics

The Powered Track Steering (PTS) Inverse Kinematics for the conversion from the trajectory controller output, [$v_{d,v}$, $\dot{\phi}_{d,v}$], to the desired translational velocities of the right and left tracks, [$V_{D,L}$, $V_{D,R}$], are defined by:

$$V_{D,L} = [1 - K.\alpha_L].[v_{d,v} - \dot{\phi}_{d,v}.\frac{D_{track}}{2}]$$
$$V_{D,R} = [1 - K.\alpha_R].[v_{d,v} + \dot{\phi}_{d,v}.\frac{D_{track}}{2}] \qquad (14)$$

where D_{track} is the distance between the tracks. In this formulation K is the slippage coefficient (a constant) that is set to $1/3$ in the system. α_L and α_R hold either one or zero, representing slippage or non-slippage situations. The slippage mechanism is explained in detail in Section VIII.

E. Path Tracking Experimental Results

Figures 13-b, -c and -d show the result of the trajectory error generation for a typical path segment change on the 1/14.2 scale excavator. In this experiment the vehicle reaches the end of a segment at time= 4.0 seconds, location A(x_w=$75cm$, y_w=$50cm$). As soon as the controller receives the next path segment, with an ending point at location B(x_w=$140cm$, y_w=$0cm$), a large amount of error in $x_{v,e}$, $y_{v,e}$ and $\phi_{v,e}$ occurs. During the next 16 seconds the controller reduces these values to zero, meeting the desired end point at B.

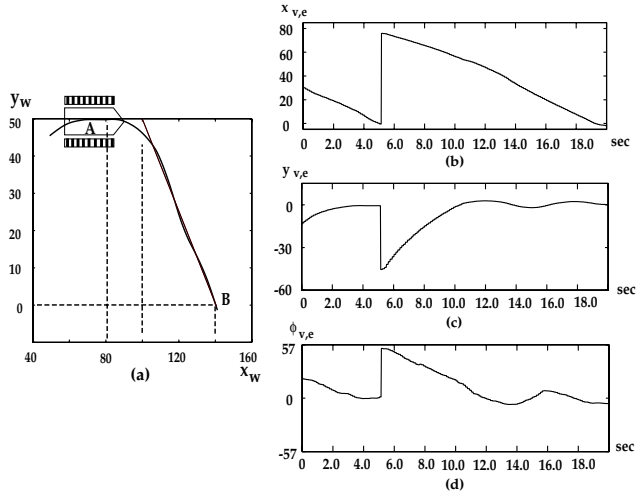


Fig. 13. Trajectory error vector generation during a path segment change. a) Excavator position; b) Position error in x; c) Position error in y; d) Orientation error.

V. Cross-Coupling Motion Control

A tracked vehicle, such as an excavator moves by the rotation of two steel or rubber tracks that are driven and controlled by two independent motors. The steering action for such a vehicle is accomplished by the difference in speed of the two tracks. The velocities for such a vehicle can be

expressed by [27]:

$$\dot{x} = \frac{v_R + v_L}{2} sin\theta \tag{15}$$

$$\dot{y} = \frac{v_R + v_L}{2} cos\theta \tag{16}$$

$$\dot{\theta} = \frac{v_R - v_L}{D_{track}} \tag{17}$$

where x, y and θ show the position and the heading of the vehicle in the world coordinates, $\dot{x}$ and $\dot{y}$ describe the translational velocities, and $\dot{\theta}$ represents the angular velocity. The linear velocities of the left and right tracks are represented by v_L and v_R and D_{track} shows the distance between the two tracks (for our full scale excavator, a Takeuchi TB035, it measures as 1.275m).

Several external and internal sources can affect the accuracy of the vehicle's motion. Overall motion error can originate from orientation and tracking errors. The heading error describes the robot's orientation error, while the tracking error shows the distance between the actual and desired vehicle positions. The heading error is the most disturbing, since it can increase the tracking error over time. Also, since each track works in an individual loop and receives no information about the other, when a disturbance happens in one loop and causes an error in the motion of the vehicle, it is corrected only in its own loop, while the other loop carries on as before. This lack of coordination can increase the heading error, and therefore, the overall translational accuracy of the resultant path. The cross-coupling controller regulates the orientation error $e\theta$ of the vehicle by exchanging feedback information between the two control loops [17].

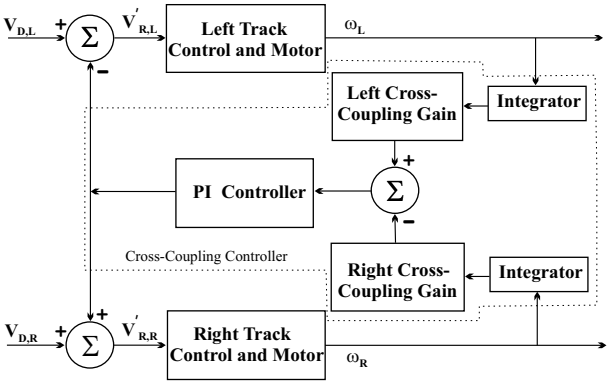


Fig. 14. The block diagram of the Cross-Coupling Motion Controller.

The idea of cross-coupling control is based on calculation of the actual error, multiplying that by a controller gain, and feeding this error back to the individual loops [28] [29], shown in Figure 14. In this figure, each of the integrator blocks calculates the linear track position in meters (taking care of unit conversions from the flowmeters). The difference in left and right track position is then calculated and passed to a PI controller. This controller ensures that the machine has zero steady-state orientation angle error assuming straight line track segments.

VI. Track Control and Motor

In the Takeuchi excavator, track motor speed can be manually controlled by foot-pedal-actuated hydraulic pilot valves. In order to permit either manual or computer control of these valves, electrohydraulic pilot valves and small hydraulic cylinders were added to the foot pedal linkage. The added pilot valves and cylinders, Takeuchi pilot valves, and track motors were considered as separate subsystems (Left and Right Track Control and Motor blocks) and a transfer function was experimentally obtained for each block. A first order transfer function (between input speed in degrees per second and output speed in degrees per second) with a pure delay was obtained for each block by fitting the subsystem experimental step response. The transfer function for left and right tracks were found to be:

$$G_L(s) = \frac{3.1}{s + 3.3} . e^{-0.2s} \tag{18}$$

$$G_R(s) = \frac{6.3}{s + 6.7} . e^{-0.2s} \tag{19}$$

For units compatibility with Figure 1 and Figure 14, an additional multiplicative constant is required to convert the reference inputs in m/sec to ^{o}/sec into each of the above transfer functions.

VII. Vision-Based Motion Tracking

In related work, a vision based 3D trajectory tracking system suitable for an autonomous robot vehicle [30] is under development here. The system includes a camera head with three on-board CCD cameras, which can be mounted anywhere on the mobile vehicle. For this work, the camera looks down and straight ahead of the excavator. By processing consecutive trinocular sets of precisely aligned and rectified images, the local 3D trajectory of the vehicle in an unstructured environment can be tracked. The system does not rely on any prior knowledge of the environment or any specific landmark in the scene. Further, the scene is mostly constructed of rigid objects, although if there are a few small moving objects the system still relies on static information. The motion of the robot is also assumed to be limited in acceleration. This allows the feature search techniques to work on a small and predictable range of possible matches. The system consists of the following components:

1) Feature Extraction: meaningful features from the scene that can be tracked over a sequence of frames are detected.
2) Stereo Vision: a 3D representation of the extracted features within the scene is obtained from a trinocular set of stereo images.
3) Feature Tracking: the features are matched using a multi-stage matching process.
4) Motion Estimation: the relative motion of the camera is estimated in an absolute reference frame.
5) Position Refinement: the 3D world feature locations are refined by combining all previous geometric measurements of the same features.

Each one of these components is considered in more detail below.

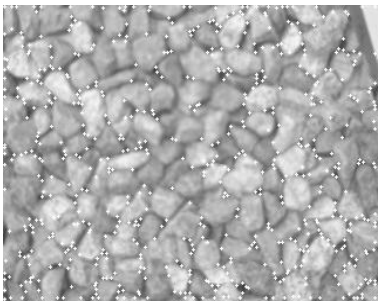


Fig. 15. A typical ground image with corners shown with bright crosses.

1) Feature Extraction: Although globally all the points in a scene convey some information about the motion, locally not all the pixel correspondences in the scene image carry valuable motion information. For example, edges, occlusions or areas of uniform intensity, can at best locally convey partial information about the motion. Also processing the entire existing image pixels is a time consuming process and includes ambiguity. For these reasons, it was decided to work with discrete points, interest points, of the scene with maximum information content that are invariant with respect to scale, rotation and point of view. Many different interest point detectors exist in the literature. However the performance of each of these methods differs from the others based on the quality of the localization, robustness with respect to noise and illumination changes, and the detection efficiency. Schmid and Mohr [31] compare and evaluate several interest point detectors based on their repeatability and information content. They show that Harris and Stephens corner detector [32] outperforms other methods with a higher geometric stability and a larger independency from imaging conditions.

Therefore, Harris and Stephens' corner detector is implemented that involves shifting a circular patch of the image in different directions. If the patch includes a corner, then shifting along all directions results in large changes. Therefore a corner can be detected when a minimum number of changes produced by any of the shifts is large enough:

$$E(x,y) = W_{u,v}|I_{x+u,y+v} - I_{u,v}|^2 \tag{20}$$

$I_{u,v}$ presents the image intensity value at point (u,v) and x and y introduce the shift amount of the circular window $W_{u,v}$

$$W_{u,v} = e^{-\frac{u^2+v^2}{\sigma^2}} \tag{21}$$

Here σ is a smoothing factor and it affects the quality of the corners found. Each corner's quality is measured from a corner response R,

$$R = Det(M) - K(T_r(M))^2 \tag{22}$$

Where,

$$M = \begin{bmatrix} X^2 \otimes W & XY \otimes W \\ XY \otimes W & Y^2 \otimes W \end{bmatrix}, \quad \text{and} \quad X \approx \frac{\partial I}{\partial x} \quad Y \approx \frac{\partial I}{\partial y} \tag{23}$$

Figure 15 shows corner detection results for a sample image.

2) Stereo Vision: The 3D world coordinates (X,Y,Z), relative to the camera for each corner are computed using a stereo algorithm. The camera system captures a set of three images which are precisely aligned horizontally and vertically [33]. Candidate feature correspondences for the overlapping regions in the three stereo images are found and the measure of Normalized Sum of Square Differences are computed for each pair of match candidates. Then, the best match candidate is found by disparity sum minimization using the multiple-baseline algorithm [34]. In addition to the epipolar constraint, agreement between the horizontal and vertical disparities is employed. This constraint eliminates unstable features, particularly those due to shadow effect. For areas of the reference image that are common in either the horizontal or vertical stereo images, the Fua [35] method is employed. Although construction of the depth is possible with just two stereo images, the use of three images enhances the accuracy of the depth and the estimated motion by eliminating invalid match candidates. The X and Y coordinates are estimated using the corresponding disparities, horizontal for X and vertical for Y. The Z value is computed using the average values of the two similar horizontal and vertical disparities, as shown in 24.

$$z = \frac{2f.B}{(d_h + d_v)} \tag{24}$$

In this equation, d_h and d_v are corresponding horizontal and vertical disparities, B represents the separation between the stereo cameras and f is the focal length of the cameras.

3) Feature Tracking: In this section, corresponding $3D$ features are tracked from one frame (at time=t) to the next (at time=$t + \Delta t$). There is no assumption or prediction about the value or direction of the motion. Systems with extra knowledge of the motion [36] from other sensors have the advantage of being able to search over smaller neighborhoods for match correspondences. Systems with more complicated features or landmarks usually track landmarks through different frames, since detecting the landmark or model from scratch may take more time. In this approach it is not possible to track identical corners from frame to frame without detecting them in each set of new images. Therefore, for each corner a simple search routine is applied in order to find all possible match candidates in the vicinity of the predicted position in the next image frame. Accordingly, a similarity metric function, the Normalized Sum of Squared Differences (NSSD), is implemented to measure the similarity of each pair of match candidates [35].

$$S = \frac{\sum_{x=\frac{-M}{2}}^{\frac{M}{2}} \sum_{y=\frac{-N}{2}}^{\frac{N}{2}} \left((I_1 - \bar{I}_1) - (I_2 - \bar{I}_2)\right)^2}{\sqrt{\sum_{x=\frac{-M}{2}}^{\frac{M}{2}} \sum_{y=\frac{-N}{2}}^{\frac{N}{2}} (I_1 - \bar{I}_1)^2 \sum_{x=\frac{-M}{2}}^{\frac{M}{2}} \sum_{y=\frac{-N}{2}}^{\frac{N}{2}} (I_2 - \bar{I}_2)^2}} \tag{25}$$

where I_1 and I_2 present the image intensities with the average values of $\bar{I}_1$ and $\bar{I}_2$. The two corners within corresponding image search regions with the highest similarity metric, S, are considered to be identical features. In order to cover

all motions, a wide search scope is required, due to two factors. The displacement of the features between frames is affected by the feature to camera distance, the rotation and/or the translation that may have occured. This wide search scope increases the number of match candidates, elevating the possibility of false matches and influences the accuracy of the motion estimation. In order to prevent false correspondence matching, a large NSSD window of 27×27 is used.

4) Motion Estimation: Having a set of corresponding corners between each two consecutive images, motion estimation becomes the problem of optimizing a 3D transformation that projects the world corners, constructed from the first image, onto the second image [37]. Although the 3D construction of 2D features is a non-linear function, the problem of motion estimation is still well-behaved. This is because any 3D motion includes rotations and translations.

- Rotations are functions of the cosine of the rotation angles.
- Translation toward or away from the camera introduces a perspective distortion as a function of the inverse of the distance from the camera.
- Translation parallel to the image plane is almost linear.

Therefore, the problem of 3D motion estimation is a promising candidate for the application of Newton's method, which is based on the assumption that the function is locally linear. To minimize the probability of converging to a false local minimum, the outliers are found and eliminated during the iteration process.

With this method, at each iteration a correction vector x is computed that is subtracted from the current estimate, resulting in a new estimate. If $P^{(i)}$ is the vector of image coordinates (u, v) for iteration i, then

$$P^{(i+1)} = P^{(i)} - x \tag{26}$$

Given a vector of error measurements between the world $3D$ features and their projections, the x is found that eliminates (minimizes) this error.

The effect of each element of the correction vector x_i on error measurement e_i is the multiplication of the partial derivative of the error with respect to that parameter to the same correction vector; this is done by considering the main assumption, local linearity of the function

$$Jx = e \qquad \text{Where} \qquad J_{ij} = \frac{\partial e_i}{\partial x_j} \tag{27}$$

J is the Jacobian matrix and e_i presents the error between the predicted location of the object and actual position of the match found in image coordinates. Each row of this matrix equation states that one measured error, e_i, should be equal to the sum of all changes in that error resulting from parameter correction [38]. Since Equation 28 is usually over-determined, and therefore, no unique solution exists, a vector x is found that minimizes the 2-norm of the residual.

$$min\|Jx - e\|^2 \tag{28}$$

Equation 28 has the same solution as the normal equation,

$$x = [J^T J]^{-1} J^T e \tag{29}$$

Therefore, in each iteration of Newton's method, the normal Equation 29 for x using LU decomposition [39] is solved. To minimize the probability of converging to a false local minimum, outliers are eliminated during the iteration process, based on their positional error.

The most computationally expensive aspect of implementing the Newton method is the calculation of the partial derivatives, or the Jacobian matrix. The partial derivatives with respect to the translation parameters can be most easily calculated by first reparametrizing the projection equations [37]. If the vector of motion parameters is $(D_x, D_y, D_z, \phi_x, \phi_y, \phi_z)$, then the new location of projected point (x, y, z) in the subsequent image is

$$(u, v) = (\frac{f(x + D_x)}{z + D_z}, \frac{f(y + D_y)}{z + D_z}) \tag{30}$$

D_x, D_y and D_z show the incremental distances (D_i) and ϕ_x, ϕ_y and ϕ_z are rotational increments about the x, y and z. The partial derivatives in the Jacobian matrix, Equation 27, are calculated from

$$\frac{\partial u}{\partial D_x} = 1 \qquad \frac{\partial u}{\partial D_y} = 0 \qquad \frac{\partial u}{\partial D_z} = \frac{fx}{(z + D_z)^2} \tag{31}$$

$$\frac{\partial u}{\partial \phi_x} = \frac{f}{z + D_z}\frac{\partial x}{\partial \phi_x} - \frac{fx}{(z + D_z)^2}\frac{\partial z}{\partial \phi_x} \tag{32}$$

$$\frac{\partial u}{\partial \phi_y} = \frac{f}{z + D_z}\frac{\partial x}{\partial \phi_y} - \frac{fx}{(z + D_z)^2}\frac{\partial z}{\partial \phi_y} \tag{33}$$

$$\frac{\partial u}{\partial \phi_z} = \frac{f}{z + D_z}\frac{\partial x}{\partial \phi_z} - \frac{fx}{(z + D_z)^2}\frac{\partial z}{\partial \phi_z} \tag{34}$$

The partial derivatives of x, y and z with respect to counterclockwise rotation parameters ϕ (in radians) can be found in Table III. This table shows how easily and efficiently the Jacobian matrix elements in Equation 27 are computed.

TABLE III
THE PARTIAL DERIVATIVES TABLE.

	x	y	z
ϕ_x	0	$-z$	y
ϕ_y	z	0	$-x$
ϕ_z	$-y$	x	0

5) Position Refinement: Several parameters can affect system accuracy. Sensor noise and quantization associated with the image can each introduce slight displacements at feature locations within the image. Furthermore, such inaccuracies can lead to faulty match correspondences between frames. As the mobile robot navigates in its environment, most of the features fall into the camera field of view for a period of several frames. Detection of a feature in each frame by itself provides additional information about that feature. Also, as the camera becomes closer to a feature, or as the features move from the image sides to its center, the 3D accuracy of the feature can improve dramatically. The second improvement is due to the fact that camera images are more distorted near the corners of an image than at the center. Therefore, combining the measurement for a feature with all its previous associated information reduces the uncertainty of that feature.

In this system, each frame, within which a feature is detected, gives an additional measurement for the location of that feature. Therefore, a positional covariance is also associated with each observed feature using a Kalman filter generation. Each filter is updated using new information for the same feature over time. The Kalman filter provides a means to combine these noisy measurements to form a continuous estimate of the current location of the feature. Each point in the world space is associated with a Kalman filter, and updated using new motion information. This process increases the accuracy of location information of the feature points in the world space. A Kalman filter is implemented in a similar fashion to Shapiro's method [40].

This formulation is recursive and the least squares estimate of the world feature position w, and its covariance C, are given recursively by

$$w_i^{-1} = w_{i-1}^{-1} + A_i^T V_i^{-1} A_i \tag{35}$$

$$x_i = x_{i-1} + k_i(b_i - A_i x_{i-1}) \tag{36}$$

$$k_i = w_i A_i^T V_i^{-1} \tag{37}$$

In these equations i represents the frame number. w_i is the uncertainty in the estimation of x corresponding to frame i, k_i denotes the filter gain, b_i indicates the current measurement of the feature, V_i represents the covariance matrix of the errors and A_i is the identity matrix. To prevent bias from distant features, working in the disparity space with axes that are the current image plane coordinates and the corresponding feature disparity [41] is implemented. Tracking of features are maintained for a while, even if they move out of the camera's field of view. However, if a feature is not seen in the last 6 consecutive frames it will be retired. Obviously, the error vector for any given measurement relies on the relative accuracy of that measurement. The corner detector's accuracy, which is related to the accuracy of the stereo system, originally 2 pixels, is improved by fitting a sub-sample estimator [42]. It is a simple quadratic estimator that locates the corner within a pixel. The method uses neighboring intensity values and fits a second order curve on the corner and its two neighbors. Since the depth construction is very sensitive to noise, this estimator improves the accuracy of the depth construction significantly.

6) Experimental Results: It is important to point out that in this system the vision sub-system is used for finding the relative motion between each two consecutive frames and not the overall trajectory of the vehicle in the global coordinate system. The Kalman filter helps to reduce the error in the vehicle's trajectory over time and if the vehicle traverses around in a bounded environment, it helps to reduce the positional error of world features as they may be viewed several times from different distances. The more accurate 3D positional information of world features, the more accurate motion estimation between every two frames.

The performance of the vision system was examined by comparing the actual position of the moving camera platform and the camera-estimated position, Figure 16. Here motion is determined while the vision system moves from a starting point A to a point D. The corresponding points (A - D) in the actual and the estimated trajectory are marked with stars. The vision system estimates all 6 variables of the 3D motion, although the physical experiment requires only 3 variables (x, y and the vertical axis orientation ϕ_z).

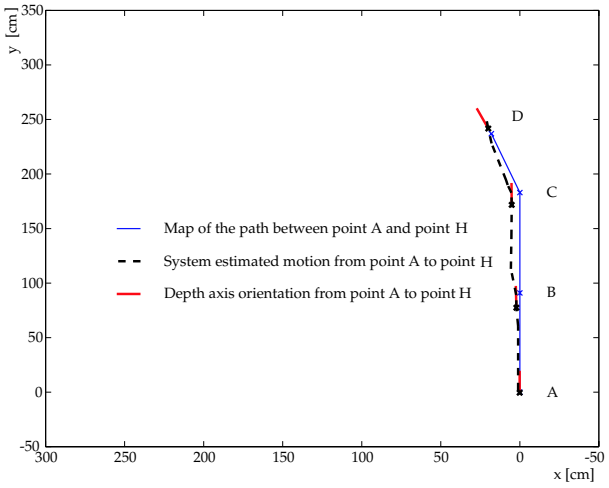


Fig. 16. The actual and estimated path using the vision system.

From this experiment, it was concluded that translational error in the worst case, is less than 5% for a typical rotational error is limited to a value of less than 1% for a 30^o rotation. Although this error is large, it is important to note that the accuracy of the slippage detection system is only dependent on the accuracy of the estimated local motion between every two consecutive frames. The translational and orientational errors for two consecutive frames are within a few percent, 5%, of the motion.

The improvement obtained by using a Kalman filter was next examined. In this experiment 75 sequential frames were processed with and without Kalman filtering. As can be seen from Figure 17, the Kalman filter significantly improves the accuracy.

For this application, the frame-to-frame accuracy is most important as that is what is used to determine slip. The camera was not used to measure world position at this time.

VIII. Slippage Controller

One major problem with tracked robots such as excavators is the track slippage during sudden starts or stops, and over various types of surfaces. Slippage usually occurs when one or both tracks lose traction with the ground surface, which makes the readings of the dead-reckoning position odometer erroneous. Moreover, when different amounts of slippage occur between left and right tracks, the tracked vehicle does not follow the appropriate curvature of the desired path. Therefore a path tracking error, and/or a position estimation error, again occurs.

With the use of only local, dead-reckoning sensors, the interaction between machine tracks and the ground, and thus slippage, cannot be detected. To overcome this problem, a novel, vision-based 3D motion tracking system is integrated with the existing excavator sensors.

Slippage is now detected by comparing the track distance traveled in a specified time period, as measured by the track

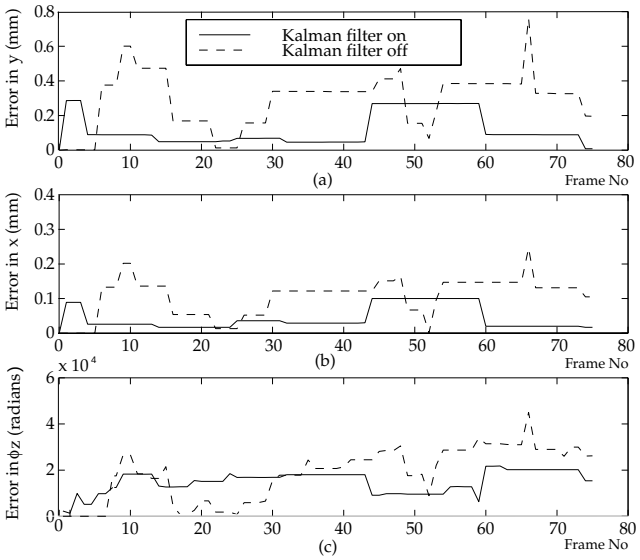


Fig. 17. Motion estimation error improvement with Kalman filtering scheme.

flow sensors, with that estimated by the vision system. Instead of relying on the dead-reckoning sensor for measuring the traveled distance, a flowmeter sensor is used. This sensor is used because readings of odometers for outdoor uneven and rough surfaces are often not very reliable.

A. Kinematic Equations of Motion

Figure 18 displays the excavator coordinate system as it moves around in two consecutive frames $k-1$ and k. The global X and Y world coordinates are shown by the axis. The local x and y excavator coordinates are the forward direction of the mobile robot and the direction perpendicular to this, in the right hand sense. The heading direction ϕ is defined as the angle between the local x axis and the global X axis. $\Delta\phi$ and ΔD define the change in heading angle and the distance traveled by the center of the machine between consecutive sampling times.

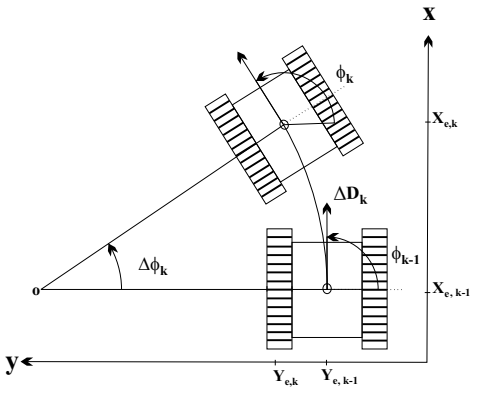


Fig. 18. Euclidean Coordinate System.

At each sampling instant, measures of the incremental distance traveled by each track in the sample time are obtained from the flowmeters, as $\Delta d_{R,flow}$ and $\Delta d_{L,flow}$ for the right and left tracks. The displacement of the center of the machine and the change in heading angle can be calculated by the relations

$$\Delta D = \frac{\Delta d_{R,flow} + \Delta d_{L,flow}}{2} \tag{38}$$

$$\Delta\phi = \frac{\Delta d_{R,flow} - \Delta d_{L,flow}}{D_{track}} \tag{39}$$

where D_{track} represents the distance between the two tracks. In order to calculate the location of the excavator at time k, it is assumed that the speed of the tracks remains constant for the entire time between each two samples. It is also assumed that the machine follows a path comprised of an arc of a circle. Note that a straight line is just an arc with an infinite radius. The location of the mobile robot at time k can be given by equations [27]

$$X_k = X_{k-1} + \frac{sin(\Delta\phi_k/2)}{\Delta\phi_k/2}\Delta D_k cos\left(\phi_{k-1} + \frac{\Delta\phi_k}{2}\right) \tag{40}$$

$$Y_k = Y_{k-1} + \frac{sin(\Delta\phi_k/2)}{\Delta\phi_k/2}\Delta D_k sin\left(\phi_{k-1} + \frac{\Delta\phi_k}{2}\right) \tag{41}$$

$$\phi_k = \phi_{k-1} + \Delta\phi_k \tag{42}$$

B. Slippage Detection

The individual track slippage condition is detected by comparing the track distance traveled during a time period, as measured by the track flowmeter sensor, with that calculated by the vision system. The derivations of the left and right track distances from the vision-based motion tracking system estimations, as well as the equation for track slippage, are presented in this section.

At each time, the vision system provides six motion parameters that represent the three translations along and three rotations around axes $(\Delta X, \Delta Y, \Delta Z, \Delta\phi_x, \Delta\phi_y, \Delta\phi_z)$. In this work, only ΔX, ΔY and $\Delta\phi_z$ are of interest, since these were the only motions measurable by the individual track hydraulic flowmeters. Figure 19 illustrates the motion of the center of the excavator and the motion of the camera system between two image frames. In this figure, the center of excavator and the center of the camera at two consecutive frames, $k-1$ and k, are given by $(X_{e,k-1}, Y_{e,k-1})$, $(X_{e,k}, Y_{e,k})$, $(X_{c,k-1}, Y_{c,k-1})$ and $(X_{c,k}, Y_{c,k})$ in the world coordinate system.

Assuming that the initial location of the excavator is $(0,0)$,

$$X_{e,k-1} = 0 \tag{43}$$

$$Y_{e,k-1} = 0 \tag{44}$$

The motion of the excavator and camera from frame $k-1$ to k in the world coordinate system can be represented in terms of the excavator center and its offset from the camera center by:

$$\Delta X_e = X_{e,k} - X_{e,k-1} = X_{e,k} \tag{45}$$

$$\Delta Y_e = Y_{e,k} - Y_{e,k-1} = Y_{e,k} \tag{46}$$

$$\Delta X_c = X_{c,k} - X_{c,k-1} = \Delta X_{vision} \tag{47}$$

$$\Delta Y_c = Y_{c,k} - Y_{c,k-1} = \Delta Y_{vision} \tag{48}$$

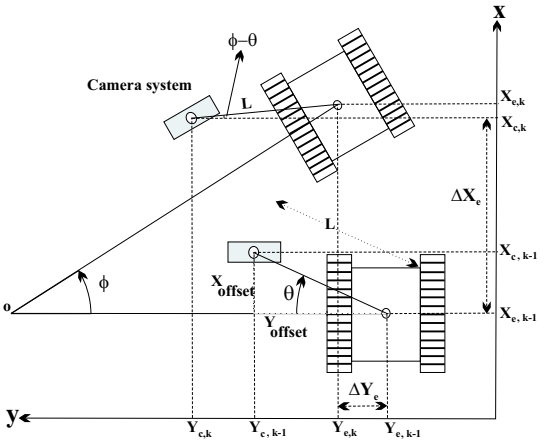


Fig. 19. Excavator and camera system in world coordinate system.

The coordinate of the camera center can be expressed by:

$$\Delta X_{c,k-1} = X_{e,k-1} + X_{offset} = X_{offset} \quad (49)$$

$$\Delta Y_{c,k-1} = Y_{e,k-1} + Y_{offset} = Y_{offset} \quad (50)$$

$$\Delta X_{c,k} = X_{e,k} + Lsin(\phi - \theta) \quad (51)$$

$$\Delta Y_{c,k} = Y_{e,k} + Lcos(\phi - \theta) \quad (52)$$

$$L = \sqrt{X^2{}_{offset} + Y^2{}_{offset}} = 0.891m \quad (53)$$

The motion of the camera can be rewritten as:

$$\Delta X_c = \Delta X_{vision} = X_{e,k} + Lsin(\phi - \theta) - X_{offset} \quad (54)$$

$$\Delta Y_c = \Delta Y_{vision} = Y_{e,k} + Lcos(\phi - \theta) - Y_{offset} \quad (55)$$

Changes in the position of the excavator can be expressed by the estimated motion of the vision system as:

$$X_{e,k} = \Delta X_{vision} + X_{offset} - Lsin(\Delta\phi_{vision} - \theta) \quad (56)$$

$$Y_{e,k} = \Delta Y_{vision} + Y_{offset} - Lcos(\Delta\phi_{vision} - \theta) \quad (57)$$

With the assumption of a constant speed for the excavator between two consecutive frames, the vision system's estimated parameters, ΔX_{vision}, ΔY_{vision} and $\Delta\phi_{vision}$, are used to compute the translational and rotational speed of the excavator.

$$V_{excav,vision} = \frac{\sqrt{\Delta X^2_{vision} + \Delta Y^2_{vision}}}{\Delta T} \quad (58)$$

$$\dot{\phi}_{excav,vision} = \frac{\Delta\phi_{vision}}{\Delta T} \quad (59)$$

Here ΔT represents the time between the two frames, 50ms. Using the same kinematics equations developed in 38 and 39 for vision, the individual left and right track distances, measured by the vision system can be estimated by:

$$\Delta d_{L,vision} = V_{excav,vision}\Delta T - \frac{D_{track}}{2}\dot{\phi}_{excav,vision}\Delta T \quad (60)$$

$$\Delta d_{R,vision} = V_{excav,vision}\Delta T + \frac{D_{track}}{2}\dot{\phi}_{excav,vision}\Delta T \quad (61)$$

During a typical instance of slippage, the distance measured by the respective track hydraulic flowmeter is larger than that estimated by the vision system. The flowmeter readings for individual left and right tracks provide $\Delta d_{L,flow}$ and $\Delta d_{R,flow}$. The track slippage, therefore, can be defined by:

$$Slip_i = \frac{\Delta d_{i,flow} - \Delta d_{i,vision}}{\Delta d_{i,vision}} \times 100 \quad (62)$$

Here the subscript i can be either L or R for the right or left track respectively. As the slippage value varies in magnitude between 0% to 100%, it represents conditions of zero track slippage to total track slippage.

$$\alpha_i = \begin{cases} 0 & \text{if } Slip_i < 20\%, \\ 1 & \text{if } Slip_i \geq 20\%. \end{cases} \quad (63)$$

During a slippage condition, coefficients of $Slip_L$ and $Slip_R$ are non-zero values. Coefficients α_L and α_R are then set to 1 using Equation 63. α_L and α_R are passed to the path tracking controller as shown in Figure 7. As soon as a slippage is detected in the path tracking controller, the destination speeds of the left and right tracks ($V_{D,L}$ and $V_{D,R}$) are reduced as represented in Equation 14, until either the slippage condition is removed or the vehicle stops. If the vehicle stops, then the operator is notified for further inspection.

These two values are used as scale factors in the PTS Inverse Kinematics equation (14) to decrease the corresponding values of $V_{D,L}$ and $V_{D,R}$.

IX. Experimental Results on Takeuchi Excavator

A Takeuchi TB035 excavator formed the target testbed machine (Figure 20). The computing platform on the Takeuchi was comprised of a VME-bus based card cage behind the operator's seat. The system was powered from the on-board 12V power supply by a 12V DC to 110V AC inverter. The on-board CPU was a SPARC 1E CPU running the VxWorks real-time operating system. Analog data was handled by a multichannel A/D board, and a D/A board. The on-board computer ran the path-planner and the machine control software at a control interval of 50 ms. A remote UNIX host machine (SPARC 5) was connected to the on-board system through an ethernet link. This host computer downloaded software initially to the on-board machine and logged data using the Stethoscope real-time utility. Stethoscope worked in conjunction with VxWorks to gather specified data variables in real-time.

Due to the computational limitations of the on-board processor, the image processing of the on-board trinocular stereo camera (Triclops) was carried out by a remote PC-based processor reporting position coordinates to the on-board processor via ethernet.

The ability of the path planner to integrate with the path tracking controller was first evaluated. This experiment was carried out on a flat concrete laboratory floor with a 5m by 5m area marked off in 1m by 1m squares. Obstacles were located in some of the squares and the locations of these were entered into the path planner. The path planner weight λ_2 was set to 0. A human interface allowed the operator to enter a goal position (x,y) and to observe the paths found by the planner. The locations of two sets of obstacles and the paths found are shown in Figure 21. A calibrated camera placed

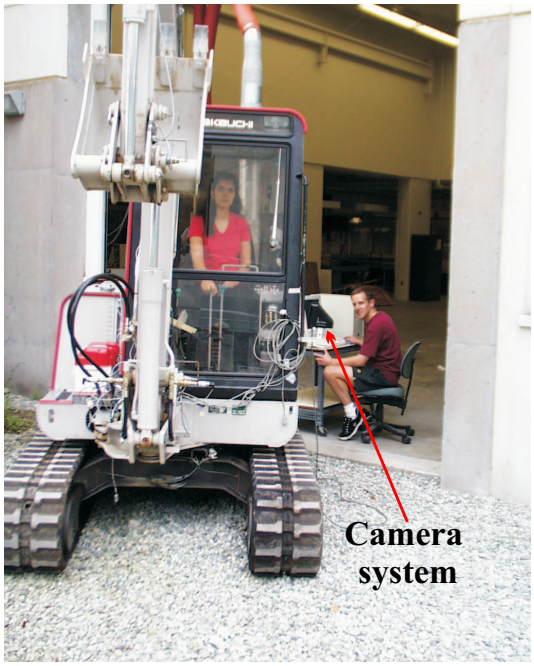


Fig. 20. The Takeuchi TB035 tracked excavator.

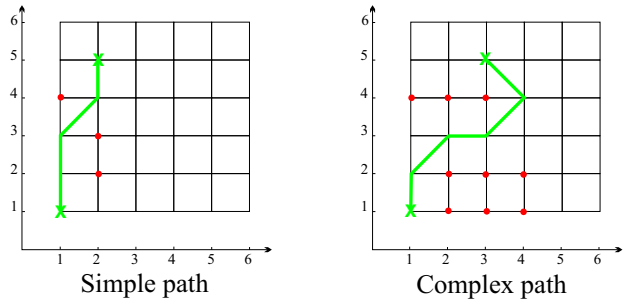


Fig. 21. Solution paths for different cases of the configuration space and goal positions.

in the ceiling of the laboratory measured the location of the centroid of three lights located on the roof of the excavator cab to an accuracy of less than 7 cm at any point during the motion of the machine along the complete path. This camera was only used for reporting positioning errors. Track odometry feedback was used in the excavator controller. Table IV shows the resulting errors as reported by the vision system.

Next, the contribution of the cross-coupling controller was evaluated. Figure 22 shows the heading angle of the vehicle with and without the cross-coupling controller [43]. When the CCMC was disabled, the heading angle, and thus heading error, constantly increased once the tracks reached their steady state value. When the CCMC was enabled, the heading angle reached a steady state value of zero degrees about 4s after the step change occurred in the track reference speeds. The divergence of the heading angle for the case the CCMC was disabled was a direct result of the fact that each of the individual track controllers acted independently of each other. Thus, with an unbounded heading angle and heading error, the path error incurred by the traveling excavator would also

TABLE IV

POSITION ERRORS MEASURED WHILE RUNNING THE SYSTEM.

	x error[cm]	y error[cm]	ϕ error[^{o}]
Simple path no. 1	4	10	-3
Simple path no. 2	3	7	-1.5
Simple path no. 3	5	10	-3
Complex path no. 1	12	14	-1
Complex path no. 2	12	7	-3
Complex path no. 3	10	15	-4.5

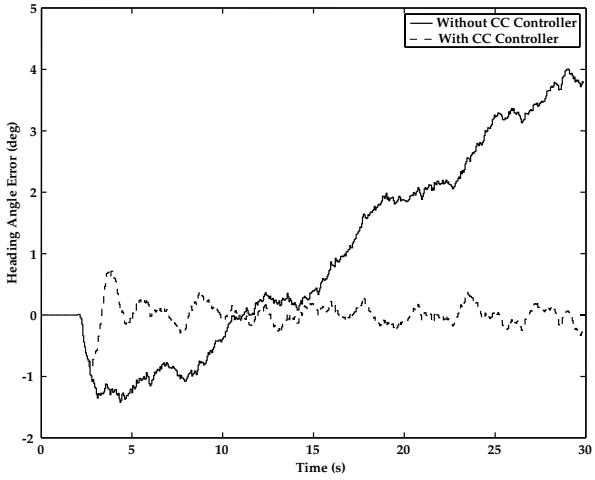


Fig. 22. Heading error with and without cross coupling controller.

be unbounded as shown in Figure 23. Also, in Figure 23 the

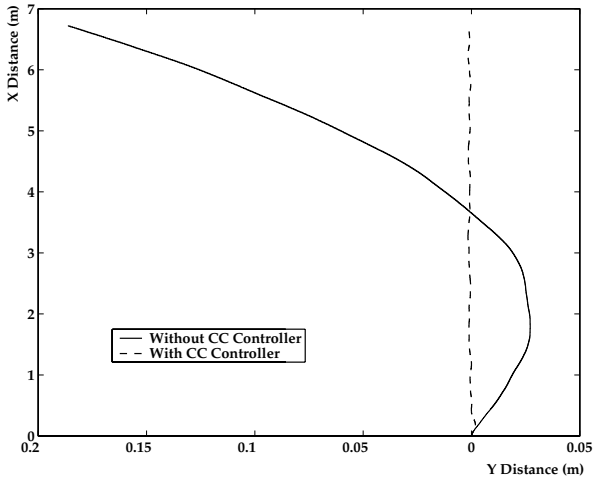


Fig. 23. Path driven for commanded straight line with and without cross coupling controller.

correct path when the CCMC was enabled is shown in dotted line for a 7 meter long path.

The performance of the slippage controller is represented through the next experiment. A 25% slippage was simulated in the machine by slowing down the right track by 25% but compensating its flowmeter to keep reporting the original speed. The track disturbance happens between times $t = 20s$ and $t = 25s$ for a 25% right track slippage. Figure 24

represents the computed slippage values in this example. At time $t = 27s$ the slippage is detected to be larger than the threshold, set for the system to 20%. Figures 25 and 26 show

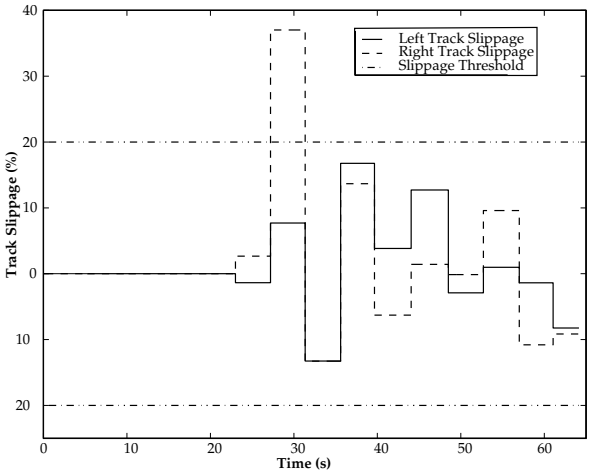


Fig. 24. Track slippage values during slippage of the right track.

the correction procedures on the left and right tracks.

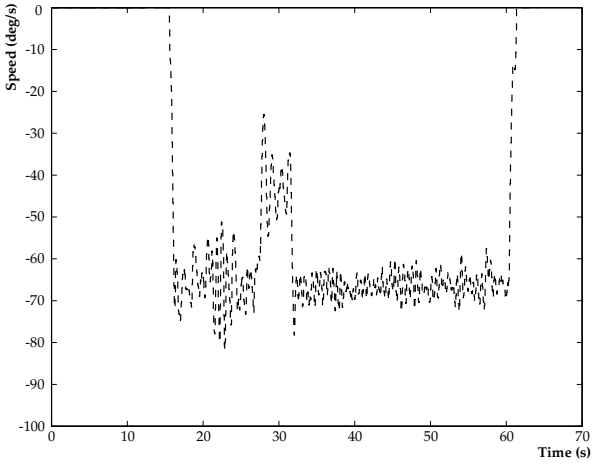


Fig. 25. Track speed response of the left track during the slippage of the right track.

As can be seen in Figure 24 a right track slippage of 37% was detected at $t = 27s$. Figure 25 and 26 show that the left and right track speeds were decreased by a factor of $1/3$ where $\alpha_L = \alpha_R = 1$. For all subsequent detection times ($t = 31.35s$ and etc.) the slippage value was below the minimum slippage threshold value of 20%, and the track reference speed scaling factors were set to $\alpha_L = 0$ and $\alpha_R = 0$. The frequency of the slippage detection mechanism is highly dependent on the speed of the vision system. Therefore a faster vision system enables the detection and correction of the slippage at an earlier time, allowing a lower slippage threshold.

X. Conclusion

Motivated by environmental considerations in off-road applications of excavator-based machines, this paper has described a complete architecture for an semi-autonomous excavator (a Takeuchi TB035) that employs a path planner to find an obstacle-free path that will trade off path length against soil damage, a path tracking controller employing fuzzy logic PD control of heading angle based on lateral track error $y_{v,e}$ and orientation error $\phi_{v,e}$, a motion controller using cross-coupling, and the novel coupling of a vision system with a path tracking controller to detect and correct excessive track slip.

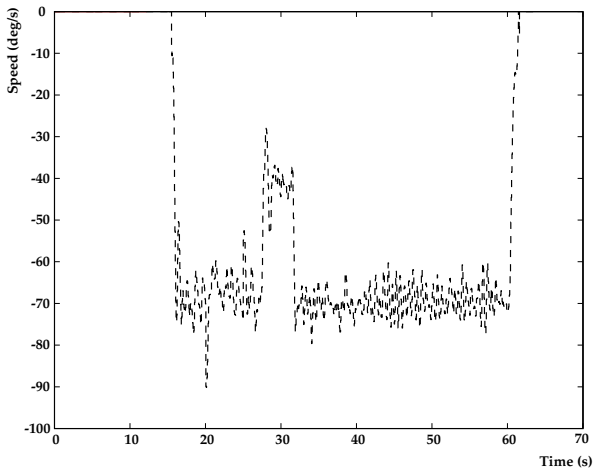


Fig. 26. Track speed response of the right track during the slippage of the right track.

The system was constrained by certain limitations. The first is that it only plans and controls piecewise linear path segments since this simplifies and speeds up the path planning. Secondly the vison system could not be used to estimate machine position and orientation due to cumulative error in the vision system as a result of its limited angle of view. We plan to resolve these limitations can be resolved in future work.

Acknowledgment

The authors are thankful to S. Bachmann for his help with the machine hydraulic system. The authors are also grateful to the Natural Sciences and Engineering Research Council of Canada (NSERC) for its support of our IRIS NCE project (Computer-Assisted Machine Operation), and their support of NSERC Discovery Grant 4924.

References

[1] A. Hemami and M. Mehrabi, "On the steering control of automated vehicles," *IEEE Conference on Intelligent Transportation System*, pp. 266–271, 1997.

[2] S. Singh, "State of the Art in Automation of Earthmoving," *ASCE Journal of Aerospace Engineering, Vol 10, Number 4*, 1997.

[3] A. Nease and E. Alexander, "Air Force Construction Automation/Robotics," *In Proc. 10th International Symposium on Automation and Robotics in Construction*, 1993.

[4] D. H. Thompson, B. L. Burks, and S. M. Killough, "Remote Excavation Using the Telerobotic Small Emplacement Excavator," *Proceedings of the American Nuclear Society Fifth Topical Meeting on Robotics and Remote Systems, Knoxville, Tenn.*, 1993.

[5] P. D. Lawrence, S. E. Salcudean, N. Sepehri, D. Chan, S. Bachmann, N. Parker, M. Zhu, and R. Frenette, "Coordinated and Force-Feedback Control of Hydraulic Excavators," *Experimental Robotics IV, The 4th International Symposium, ISER95, Vol. 223, pp. 181-194, Stanford, California, USA*, 1995.

[6] I. S. Kweon and T. Kanade, "Extracting Topographic Terrain Features from Elevation Maps," *CVGIP: Image Understanding*, vol. 59, pp. 171–182, March 1994.

[7] J. S. B. Mitchell, "An algorithmic approach to some problems in terrain navigation," *Artificial Intelligence*, vol. 37, no. 1-3, pp. 171–201, 1988.

[8] Z. Alexander, "A Mobile Robot Navigation Exploration Algorithm," 1992.

[9] M. Mehrabi, A. Hemami, and R. Cheng, "Analysis of Steering Control in Vehicles with Two Independent Left and Right Traction Wheels," *Fifth International Conference on Advanced Robotics, pp. 1634-1637, Pisa, Italy*, 1991.

[10] Y. Zhang, D. Hong, J. Chung, and S. Velinsky, "Dynamic Model Based Robust Tracking Control of a Differentially Steered Wheeled Mobile Robot," *Proceedings of the American Control Conference, pp. 850-855, Philadelphia, Pennsylvania*, 1998.

[11] K. Koh and H. Cho, "A Smooth Path Tracking Algorithm for Wheeled Mobile Robots with Dynamic Constraints," *Journal of Intelligent and Robotic Systems, Vol. 24, pp. 367-385*, 1999.

[12] S. Singh and D. Shino, "Position Based Path Tracking for Wheeled Mobile Robots," *IEEE/RSJ International Workshop on Intelligent Robots and Systems, pp. 386-391, Tsukuba, Japan*, 1989.

[13] C. Samson, "Control of chained systems: application to path following and time-varying point-stabilization of mobile robots," *IEEE Transactions on Automatic Control*, vol. 40, pp. 64–77, 1995.

[14] D. del Ro, Jimnez, Sevillano, Vicente, and C. Balcells, "A path following control for Unicycle robots," *Journal of Robotic Systems*, pp. 325–342, 2001.

[15] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," *IEEE Int. Conf. Robotics and Autom*, pp. 384–389, 1990.

[16] D. Kusalovic, "Navigation System for Semi-Autonomous Tracked Machines," *MASc Thesis, University of British Columbia*, 1998.

[17] Y. K. L. Feng and J. Borenstein, "Cross-Coupling Motion Controller for Mobile Robots," *IEEE Control Systems Magazine*, pp. 35–43, 1993.

[18] *Terrain Resource Information Management Program*. Ministery of Sustaneable Resourceable Management, British Columbia, Canada. `http://srmwww.gov.bc.ca/bmgs/trim/trim/trim_overview/trim_program.htm`.

[19] R. Keeney and H. H. Raiffa, *Decisions with Multiple Objectives: Preference and Value Tradeoffs*. John Wiley & Sons, 1976.

[20] J. Michell, "An Algorithmic Approach to Some Problems in Terrain Navigation," *Artificial Intelligence, 37, 171–197*, 1988.

[21] J. Benton, S. Iyengar, W. Deng, N. Brener, and V. Subrahmanian, "Tactical Route Planning: New Algorithms for Decomposing the Map," *IEEE, Proceedings of the IX Conference on Mobile Robotics*, 1995.

[22] J. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1993.

[23] L. Bolc and J. Cytowski, *Search Methods for Artificial Intelligence*. Academic Press, 1992.

[24] A. N. Y. Kanayama and C. Lelm, "A locomotion control method for autonomous vehicles," *IEEE Conference on Robotics and Automation*, pp. 1315–1317, 1988.

[25] P. Jacobsen, "Control of Tracked Vehicle," *Masters Thesis, Technical University of Denmark, performed at University of British Columbia*, 1995.

[26] M. J. T. Ross and N. Vadiee, *Fuzzy Logic and Control*. Prentice Hall, 1993.

[27] C. M. Wang, "Location Estimation and Uncertainty Analysis for Mobile Robots," *IEEE Conference on Robotics and Automation*, pp. 1230–1235, 1988.

[28] P. Kulkarni and K. Srinivasan, "Optimal contouring control of multi-axial feed drive servomechanisms," *ASME Journal of Dynamic systems, measurements and control*, pp. 140–148, 1989.

[29] K.Srinivasan and P.K.Kulkarni, "Cross-Coupled Control of Biaxial Feed Drive Servomechanisms," *ASME Journal of Dynamic Systems, Measurement and Control*, pp. 225–232, 1990.

[30] P. Saeedi, P. Lawrence, and D. Lowe, "3D motion tracking of a mobile robot in a natural environment," *IEEE International Conference on Robotics and Automation*, pp. 1682–1687, 2000.

[31] R. M. C. Schmid and C. Bauckhage, "Comparing and Evaluating Interest Points," *Sixth International Conference on Computer Vision*, pp. 230–235, 1998.

[32] C. Harris and M. Stephens, "A combined corner and edge detector," *Proceeding 4'th Alvey Vision Conference*, pp. 147–151, 1988.

[33] P. G. Research, "Triclops Stereo Vision System," tech. rep., Department of Computer Science, University of British Columbia, Vancouver, www.ptgrey.com, 1997.

[34] M. Okutomi and T. Kanade, "A multiple-baseline stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 353–363, 1993.

[35] P. Fua, *A parallel stereo algorithm that produces dense depth maps and preserves image features*. Machine Vision and Applications, Springer-Verlag, 1993.

[36] S. Se, D. Lowe, and J. Little, "Vision-based mobile robot localization and mapping using scale-invariant features," *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 2051–2058, 2001.

[37] D. G. Lowe, *Three-dimensional object recognition from single two-dimensional images*. Artificial Intelligence, Elsevier Science Publishers B.V. (North-Holland), 1987.

[38] D. G. Lowe, "Fitting Parameterized Three-dimensional Models to Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 441–450, 1991.

[39] W. Press, S. Teukolsk, W. Vetterling, and B. Flannery. Numerical Recipes in C: The Art of Scientific Computing, Cambridge University Press, 1992.

[40] L. Shapiro, A. Zisserman, and M. Brady, "3D Motion Recovery via Affine Epipolar Geometry," *International Journal of Computer Vision, Vol 16*, pp. 147–182, 1995.

[41] C. Harris and J. Pike, "3D positional integration from image sequences," *Image and Vision Computing*, pp. 87–90, 1988.

[42] V. Dvornychenko, "Bounds on (deterministic) correlation functions with application to registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 206–216, 1983.

[43] K. Ardron, "A low Level Motion Controller for Path Tracking Control of a Tracked Mobile Robot," *MASc Thesis, University of British Columbia*, 2002.

Parvaneh Saeedi received the B.A.Sc. degree in electrical engineering from the Iran University of Science and Technology, Tehran, Iran and the Master's degree in electrical and computer engineering from the University of British Columbia, Vancouver, BC, Canada in 1998. She is now pursuing the Ph.D. degree in electrical and computer engineering at the University of British Columbia. Her research interests include computer vision, motion tracking and automated systems for DNA image analysis.

Peter D. Lawrence received the B.A.Sc. degree in electrical engineering from the University of Toronto, Toronto, ON, Canada, in 1965, the Masters's degree in biomedical engineering from the University of Saskatchewan, Saskatoon, SK, Canada, in 1967 and the Ph.D. degree in computing and information science at Case Western Reserve University, Cleveland, OH, in 1970.

He worked as Guest Researcher at Chalmers University's Applied Electronics Department, Goteborg, Sweden between 1970 and 72, and between 1972 and 1974 as a Research Staff Member and Lecturer in the Mechanical Engineering Department of the Massachusetts Institute of Technology, Cambridge. Since 1974, he has been with the University of British Columbia and is a Professor in the Department of Electrical and Computer Engineering. His main research interests include the application of real-time computing in the control interface between humans and machines, image processing, and mobile hydraulic machine modeling and control.

Dr. Lawrence is a member of SAE and a registered Professional Engineer.

David G. Lowe is a professor of computer science at the University of British Columbia. He received his Ph.D. in computer science from Stanford University in 1984. From 1984 to 1987 he was an assistant professor at the Courant Institute of Mathematical Sciences at New York University. His research interests include object recognition, local invariant features for image matching, robot localization, object-based motion tracking, and models of human visual recognition. He is on the Editorial Board of the International Journal of Computer Vision and was co-chair of ICCV 2001 in Vancouver, Canada.

PLACE
PHOTO
HERE

Kevin Ardron received the B.A.Sc degree in Electrical Engineering from the University of British Columbia, Vancouver, BC, Canada in 1997 and the M.A.Sc degree in Electrical and Computer Engineering in 2003. His research interests include real-time systems, mobile robot control and computer vision.

Paul H. Sorensen received his B. Sc. in mathematics and physics and his M. Sc. in theoretical physics from the University of Aarhus, Denmark in 1982. He obtained a Ph.D. in theoretical high-energy physics from the University of Cambridge, United Kingdom in 1987.

He joined the faculty at the Technical University of Denmark as an assistant professor of control engineering in 1985 and became associate professor in the electrical engineering department in 1989. He was a visiting professor at the University of Reading, UK in 1993 and a visiting professor at the Mechanical and Aerospace Engineering department of North Carolina State University, USA. Currently, he is a project engineer with Creare Inc. in Hanover, New Hampshire. His interests range from control systems, signal processing to estimation theory and image processing. Dr. Sorensen is a member of IEEE Control Systems Society.