2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Visual Teach and Repeat, Repeat, Repeat: Iterative Learning Control to Improve Mobile Robot Path Tracking in Challenging Outdoor Environments

Chris J. Ostafew, Angela P. Schoellig, and Timothy D. Barfoot

Abstract— **This paper presents a path-repeating, mobile robot controller that combines a feedforward, proportional Iterative Learning Control (ILC) algorithm with a feedback-linearized path-tracking controller to reduce path-tracking errors over repeated traverses along a reference path. Localization for the controller is provided by an on-board, vision-based mapping and navigation system enabling operation in large-scale, GPS-denied, extreme environments. The paper presents experimental results including over 600 m of travel by a four-wheeled, 50 kg robot travelling through challenging terrain including steep hills and sandy turns and by a six-wheeled, 160 kg robot at gradually-increased speeds up to three times faster than the nominal, safe speed. In the absence of a global localization system, ILC is demonstrated to reduce path-tracking errors caused by unmodelled robot dynamics and terrain challenges.**

I. Introduction

Large-scale mapping and navigation by autonomous mobile robots in GPS-denied environments remains a significant challenge for robotics. In many mobile robot applications, it is adequate if not necessary, to explore and navigate the environment by creating and maintaining a network of paths analogous to migration routes or automobile roads [1]. The use and reuse of paths reduces the need for repeated application of exploratory and terrain assessing software and also provides an opportunity for learning behavior. Learning control algorithms offer tools to extract control information in situ and improve performance over time, as opposed to control algorithms that rely on significant modelling a priori [2]. In this paper, we investigate Iterative Learning Control (ILC) as an added-benefit, feedforward control for a practical, nonholonomic, mobile robot within the context of a real-time Visual Teach and Repeat (VT&R) mapping and navigation system [3].

There are several systematic benefits when employing ILC on a mobile, path-repeating robot. First, the learning algorithm reduces the design effort necessary to predict path disturbances and robot dynamics prior to deployment. In practice, ILC iteratively learns a feedforward control signal for future trials based on previous path-tracking errors. In addition to disturbances caused by unmodelled terrain topography or robot dynamics, the ILC algorithm is capable of learning new errors caused by environmental change or intentionally increased velocity. Second, by learning the system and environmental models, the ILC algorithm affords a simple and computationally light feedback controller. Third, the ILC algorithm affords minimal controller redesign or tuning when transferring to other robots, in our case two different skid-steered robots of varying dimensions and mass.

The authors are with the Autonomous Space Robotics Lab at the University of Toronto Institute for Aerospace Studies, Toronto, Ontario, M3H 5T6, Canada. Email: chris.ostafew@mail.utoronto.ca, schoellig@utias.utoronto.ca, and tim.barfoot@utoronto.ca. A supplemental video for this paper is also available at http://ieeexplore.ieee.org.

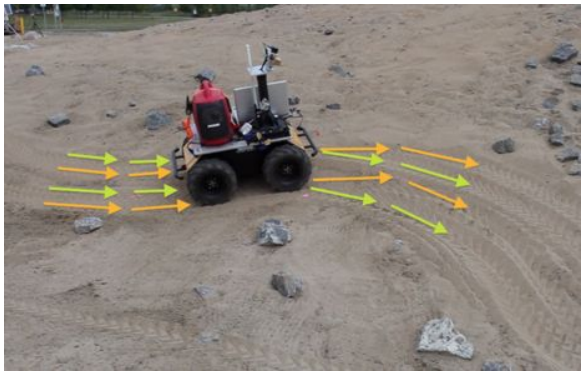


Fig. 1. Husky A200 robot tracking a path with a 5-15° side slope. The orange arrows show the current trajectory of the robot as it drives with ILC disabled. The green arrows show the path with reduced errors from when the robot was travelling with ILC enabled.

Fig. 2. ROC6 robot tracking a path at three times the nominal safe speed of 0.35 m/s. The orange arrows show the current trajectory of the robot as it drives with ILC disabled. The green arrows show the path with reduced errors from when the robot learned to travel at speed with ILC enabled.

The key characteristics of the presented work are: navigation based on vision only, a path-tracking controller comprised of feedforward ILC and feedback-linearized control, an ILC scheme parameterized by path length, and extensive outdoor experiments on two significantly different robots in unmodelled terrain and at intentionally increased velocities.

The structure of this paper is as follows. First, a brief background on ILC is given. Then we present a summary of our autonomous VT&R system (our source of localization along the path). Next, the formulation for the ILC solution

is presented. In Section V, we discuss the experimental setup and the results obtained. Finally, in Section VI, we present a brief conclusion and future work.

II. Contribution

ILC was first introduced in the literature by Arimoto et al. [4] to give systems that operate repetitively the ability to take advantage of knowledge gained during previous iterations. ILC is commonly applied to industrial systems in manufacturing, robotics, and chemical processing. However, there exist few references in the literature of applications of ILC in mobile robotics and none, to our knowledge, operating on a self-contained autonomous vehicle travelling in challenging outdoor environments.

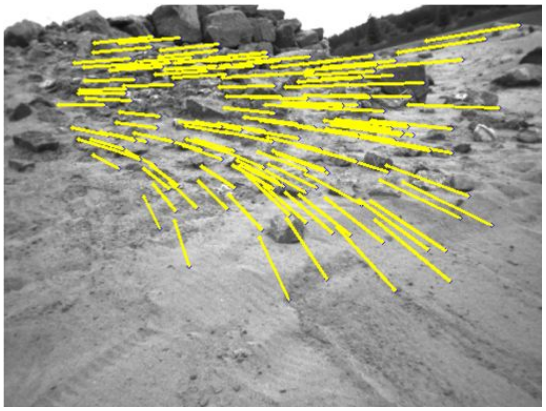
Recent sources for ILC literature include survey papers by Bristow et al. [5] and Ahn et al. [6]. In the review by Bristow et al., ILC design is decomposed into (i) Proportional-Derivative-type (PD) laws, (ii) plant inversion laws, (iii) H_∞ laws, and (iv) optimal laws. Each technique uses error information from past trials to construct a feed-forward signal that reduces the tracking error.

In this paper, we experiment with a PD-type law and demonstrate the application of ILC on two significantly different mobile robots in outdoor environments with truly unknown terrain and unknown vehicle dynamics. In previous work, Oriolo et al. [7], Kang et al. [8], and Han and Lee [9] demonstrated PD-type ILC for mobile robots on short, indoor paths less than 4 m in length using wheel odometry for localization. Chen and Moore [10] demonstrated PD-type ILC for mobile robots in simulation. In contrast, our implementation is founded on VT&R, a vision-based, on-board localization system that enables us to test on very long paths and in challenging outdoor terrain without accumulating dead-reckoning error during each trial. First, we show results from the repetition of a 40-m-long, outdoor path with sand and side-slopes by a 50 kg four-wheeled robot. Then we show the results from the repetition of a 50-m-long path by a 150 kg six-wheeled robot learning to drive quickly.

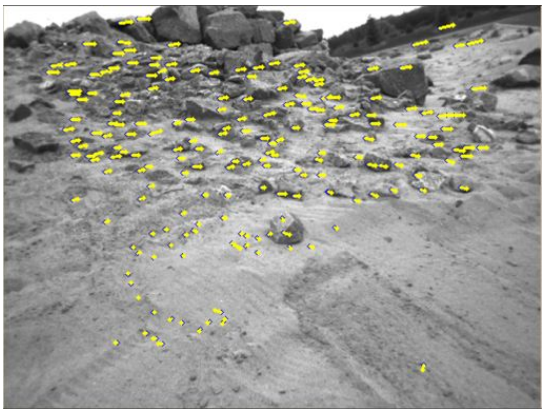
Unlike wheel-odometry-based dead-reckoning, vision-based localization depends on the camera perspective and thus relatively precise path-tracking during every trial in order to relocalize. Our PD-type ILC controller is inspired by the work of Kang et al. [8] with the addition of a feedback-linearized controller. Furthermore, our ILC feed-forward signal is spatially-indexed to facilitate integration to our VT&R system and enable changing robot speed over trials. In previous work on mobile robots [7, 8, 9, 10], the feedforward ILC signal is parameterized by time and altering the speed on successive trials is non-trivial. In our work, the robot begins an experiment at a safe low speed and is capable of iteratively learning to drive faster.

III. Visual Teach and Repeat

Localization for our mobile robot is provided by an on-board VT&R algorithm developed by Furgale and Barfoot [3] where the sole sensor is an on-board stereo camera. In the first operational phase, the robot is driven along the desired path manually by an operator or following a path provided by an on-board autonomous path planner. Localization in this initial operation is obtained relative to the robot's starting position by visual odometry (VO). In addition to the VO pipeline, path vertices are defined after each 0.2 m of travel or 3.5° of rotation by storing keyframes composed of local feature descriptors and their 3D positions. During the second operational phase, the repeat phase, the robot re-localizes against the stored keyframes thus generating feedback for a path-tracking controller (see Figure 3). Re-localization is achieved by matching feature descriptors to generate feature tracks between the current robot view and the teach-pass robot view. Outlier feature-matches are rejected using the Random Sample and Consensus (RANSAC) [11] algorithm. As long as a sufficient number of correct feature matches are made, the system generates consistent localization over trials and has the ability to support an ILC feedforward algorithm.



(a) Image taken during the first trial of experiment 1 and so is representative of a relatively large path-tracking error.



(b) Image taken during sixth trial of experiment 1 with low path-tracking error.

Fig. 3. Visual representations of re-localization in our VT&R framework. Each feature track represents the translation between a feature identified during the teach phase and re-identified during a repeat phase.

In practice, several other VT&R systems exist; however, they have not been documented to use any learning behavior for path tracking. In Šegvić et al. [12] and Krajník et al. [13], path tracking is accomplished by visual servoing. While in Marshall et al. [14] and Royer et al. [15], path tracking is accomplished by feedback linearization and chained-form conversion, respectively.

IV. Mathematical Formulation

A. Feedback Linearization

Our path-tracking control system is experimentally verified on a Clearpath Husky A200 robot (see Figure 1) and a Robosoft ROC6 robot (see Figure 2), which are both skid-steered robots. We use feedback linearization as described by Samson and Ait-Abderrahim [16] as our base path-tracking controller. Both robots are modelled as unicycle-type robots with coordinates, $\boldsymbol{\epsilon}[k] = [\epsilon_X[k]\, \epsilon_L[k]\, \epsilon_H[k]]^T$, representing the longitudinal, lateral, and heading errors at time k, respectively, and calculated relative to the nearest path vertex by Euclidean distance (see Figure 4). When the linear and angular velocities of the robot are defined as $v[k]$ and $\omega_{fb}[k]$, respectively, and the time between control signal updates is defined as Δt, the resulting error dynamics are

$$\begin{bmatrix} \epsilon_L[k+1] \\ \epsilon_H[k+1] \end{bmatrix} = \begin{bmatrix} \epsilon_L[k] \\ \epsilon_H[k] \end{bmatrix} + \Delta t \begin{bmatrix} v[k] \sin \epsilon_H[k] \\ \omega_{fb}[k] \end{bmatrix},$$

where $v[k]$ and $\omega_{fb}[k]$ are system inputs and $\omega_{fb}[k]$ is provided by the feedback controller. Then, assuming the robot forward velocity command is constant, and letting $z_1[k] := \epsilon_L[k]$, $z_2[k] := v[k] \sin \epsilon_H[k]$, and $\eta[k] := v[k] \cos \epsilon_H[k] \omega_{fb}[k]$, a new system of equations is given by

$$\begin{aligned} \begin{bmatrix} z_1[k+1] \\ z_2[k+1] \end{bmatrix} &= \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_1[k] \\ z_2[k] \end{bmatrix} + \Delta t \begin{bmatrix} 0 \\ v[k] \cos \epsilon_H[k] \omega_{fb}[k] \end{bmatrix} \\ &= \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_1[k] \\ z_2[k] \end{bmatrix} + \Delta t \begin{bmatrix} 0 \\ \eta[k] \end{bmatrix}. \end{aligned}$$

Choosing a proportional controller of the form $\eta[k] = -\gamma_1 z_1[k] - \gamma_2 z_2[k]$ with $\gamma_1, \gamma_2 > 0$ gives the stable, closed-loop system

$$\begin{bmatrix} z_1[k+1] \\ z_2[k+1] \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ -\Delta t \gamma_1 & 1 - \Delta t \gamma_2 \end{bmatrix} \begin{bmatrix} z_1[k] \\ z_2[k] \end{bmatrix}. \quad (1)$$

Solving for the feedback-linearized control input, $\omega_{fb}[k]$, given the two definitions of $\eta[k]$, we find

$$\omega_{fb}[k] = \frac{-\gamma_1 \epsilon_L[k] - \gamma_2 v[k] \sin \epsilon_H[k]}{v[k] \cos \epsilon_H[k]}.$$

With suitable calibrations and paths, this closed-loop system implemented on the ROC6 robot has been demonstrated to follow paths up to 3.2 km in length autonomously [3] at speeds up to 0.35 m/s.

B. Added-benefit ILC

In practice, increasing the forward velocity or travelling on paths with sand or side-slopes, can result in increased path-tracking errors that may put the robot in danger and that challenge the ability of VT&R to relocalize. As shown in Figure 5, ILC learns a feedforward control input, ω_{ff}, from these path-tracking errors. The ILC algorithm is able to generate a feedforward signal that anticipates and pre-emptively responds to repeated disturbances and unmodelled robot dynamics since it has access to the entire sequence of errors along the path from previous trials. Intuitively, the learned, feedforward signal amounts to steering corrections along the path.

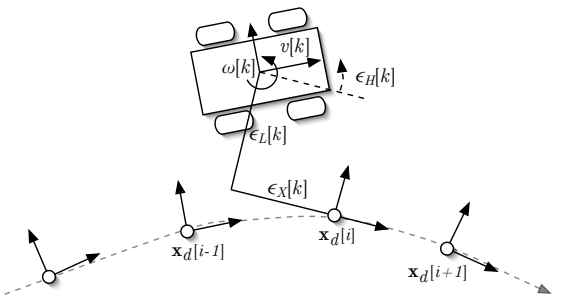


Fig. 4. Definition of robot velocities and path-tracking errors. The robot pose in 3D is projected down into the plane of the nearest pose vertex, then the longitudinal, lateral, and heading errors, $\epsilon_X[k], \epsilon_L[k], \epsilon_H[k]$, are calculated.

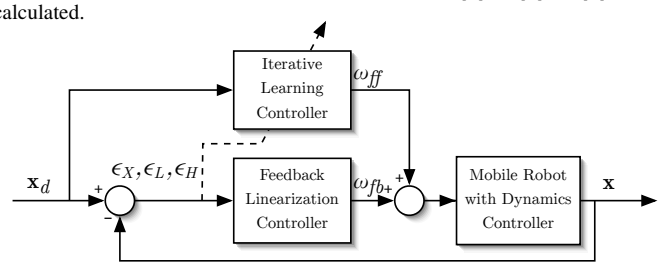


Fig. 5. The controller block diagram. The Iterative Learning Controller is in parallel to the Feedback-Linearization Controller and learns only from path-tracking errors.

Since our VT&R system is tracking a path rather than a trajectory, the feedforward commands are a function of vertex indices. We introduce k_i as the time index at which the ith vertex is reached. Then with n_i time steps, Δt, between the ith and ith$+1$ vertices so that $k_{i+1} = k_i + n_i$, and letting

$$\begin{aligned} \mathbf{A}[k_i] &:= \begin{bmatrix} 1 & n_i \Delta t \\ -n_i \Delta t \gamma_1 & 1 - n_i \Delta t \gamma_2 \end{bmatrix}, \\ \mathbf{B}[k_i] &:= n_i \Delta t \begin{bmatrix} 0 \\ v[k_i] \cos \epsilon_H[k_i] \end{bmatrix}, \\ \mathbf{z}[k_i] &:= \begin{bmatrix} z_1[k_i] \\ z_2[k_i] \end{bmatrix}, \end{aligned}$$

we model the system with an ILC feedforward signal as

$$\mathbf{z}[k_{i+1}] = \mathbf{A}[k_i]\mathbf{z}[k_i] + \mathbf{B}[k_i]\omega_{ff}[k_i].$$

Since there are K vertices in our path, we can produce $K-1$ relationships for a given path,

$$\begin{aligned} \mathbf{z}[k_1] &= \mathbf{A}[k_0]\mathbf{z}[k_0] + \mathbf{B}[k_0]\omega_{ff}[k_0], \\ \mathbf{z}[k_2] &= \mathbf{A}[k_1]\mathbf{z}[k_1] + \mathbf{B}[k_1]\omega_{ff}[k_1], \\ &= \mathbf{A}[k_1]\mathbf{A}[k_0]\mathbf{z}[k_0] + \mathbf{A}[k_1]\mathbf{B}[k_0]\omega_{ff}[k_0] + \mathbf{B}[k_1]\omega_{ff}[k_1], \\ &\vdots \end{aligned}$$

that can be organized in matrix form to produce the 'lifted form' [5] for trial j:

$$\mathbf{z}_j = \mathbf{z}_0 + \mathbf{B}_j \boldsymbol{\omega}_j,$$

where

$$\mathbf{z}_j = \begin{bmatrix} \mathbf{z}[k_1] \\ \mathbf{z}[k_2] \\ \vdots \\ \mathbf{z}[k_K] \end{bmatrix}, \quad \mathbf{z}_0 = \begin{bmatrix} \mathbf{A}[k_0]\mathbf{z}[k_0] \\ \vdots \\ \mathbf{A}[k_{K-1}] \dots \mathbf{A}[k_0]\mathbf{z}[k_0] \end{bmatrix},$$

$$\mathbf{B}_j = \begin{bmatrix} \mathbf{B}[k_0] & 0 & 0 \dots & 0 \\ \mathbf{A}[k_1]\mathbf{B}[k_0] & \mathbf{B}[k_1] & 0 \dots & 0 \\ \vdots & & & \\ \mathbf{A}[k_{K-1}] \dots \mathbf{A}[k_1]\mathbf{B}[k_0] & \dots & & \mathbf{B}[k_{K-1}] \end{bmatrix},$$

$$\boldsymbol{\omega}_j = \begin{bmatrix} \omega_{ff}[k_0] \\ \omega_{ff}[k_1] \\ \vdots \\ \omega_{ff}[k_{K-1}]. \end{bmatrix}.$$

Once in lifted form, we can introduce the error for trial j

$$\mathbf{e}_j = \mathbf{z}_d - \mathbf{z}_j,$$

where $\mathbf{z}_d$ is the desired state and is identically zero. Then

$$\mathbf{e}_j = -\mathbf{z}_0 - \mathbf{B}_j\boldsymbol{\omega}_j,$$

and

$$\begin{aligned} \mathbf{e}_{j+1} - \mathbf{e}_j &= -\mathbf{B}_{j+1}\boldsymbol{\omega}_{j+1} + \mathbf{B}_j\boldsymbol{\omega}_j \\ &\approx -\mathbf{B}_j\boldsymbol{\omega}_{j+1} + \mathbf{B}_j\boldsymbol{\omega}_j. \end{aligned}$$

To compute the feedforward control input for the next trial we use a proportional-type ILC controller of the form

$$\boldsymbol{\omega}_{j+1} = \gamma_u\boldsymbol{\omega}_j + \boldsymbol{\Gamma}_e\mathbf{e}_j.$$

with forgetting factor, γ_u, and learning gain, $\boldsymbol{\Gamma}_e$. As a result, we have that

$$\begin{aligned} \mathbf{e}_{j+1} &= \mathbf{e}_j - \gamma_u\mathbf{B}_j\boldsymbol{\omega}_j - \mathbf{B}_j\boldsymbol{\Gamma}_e\mathbf{e}_j + \mathbf{B}_j\boldsymbol{\omega}_j \\ &= \mathbf{e}_j + \gamma_u(\mathbf{e}_j + \mathbf{z}_0) - \mathbf{B}_j\boldsymbol{\Gamma}_e\mathbf{e}_j - (\mathbf{e}_j + \mathbf{z}_0) \\ &= (\gamma_u - \mathbf{B}_j\boldsymbol{\Gamma}_e)\mathbf{e}_j - (1 - \gamma_u)\mathbf{z}_0. \end{aligned}$$

Stabilization in the trial domain amounts to selecting γ_u and $\boldsymbol{\Gamma_e}$ such that the eigenvalues of $(\gamma_u - \mathbf{B}_j\boldsymbol{\Gamma}_e)$ are stable. In practice, we used the update law

$$\omega_{j+1}[k_i] = \gamma_u\omega_j[k_i] + \frac{-\gamma_L\epsilon_{Li} - \gamma_H \sin\epsilon_{Hi}}{\cos\epsilon_{Hi}}, \qquad (2)$$

where ϵ_{Li} and ϵ_{Lh} are lookahead errors,

$$\epsilon_{Li} = \frac{1}{m-1}\sum_{j=i+2}^{i+m}\epsilon_L[k_j], \quad \epsilon_{Hi} = \frac{1}{m-1}\sum_{j=i+2}^{i+m}\epsilon_H[k_j].$$

and m is the number of vertices to lookahead. In equation 2, the gains γ_L and γ_H set the rate of convergence while the forgetting factor γ_u gives the system the ability to forget nonrepetitive disturbances. Tuning begins by setting $\gamma_u = 1$ and adjusting γ_L and γ_H until a reasonable rate of convergence is found. Then γ_u is decreased to give the system robustness to nonrepetitive disturbances and changes in the environment.

Fig. 6. Experiment 1: Husky A200 path at the Canadian Space Agency's Mars Emulation Terrain in Longueuil, Québec, Canada. The terrain offers a large selection of surface topographies intended for robot mobility testing.

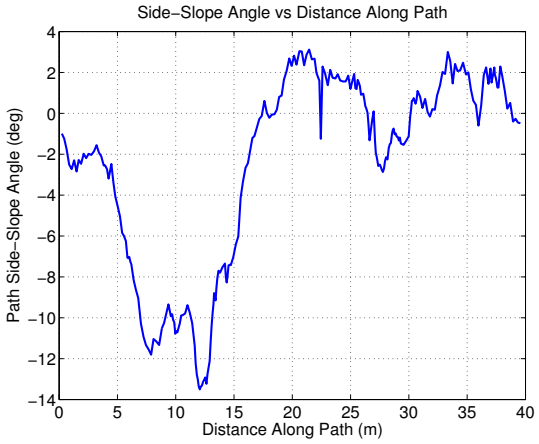


Fig. 7. Experiment 1: Husky A200 path side-slope angle. The path selected included a steep side-slope around 10-15 m into the path. Another photo of the side-slope section is shown in Figure 1.

V. Experiment

A. Overview

The experiment consisted of two parts: (i) a test with a 50 kg Husky A200 robot travelling along a 40-m-long, sandy path including an unmodelled side-slope, and (ii) a test with a 160 kg ROC6 robot sequentially navigating a 50-m-long path with speeds gradually increased to three times the speed for which the feedback-linearized controller was tuned. The side-slope test was conducted on the Canadian Space Agency's Mars Emulation Terrain (MET) in a sandy section with slopes of between 5-15° as shown in Figures 6 and 7. The speed test was conducted in the University of Toronto Institute for Aerospace Studies (UTIAS) MarsDome along a path through tight valleys and over a combination of gravel and sandy surfaces as shown in Figure 8.

In both cases, the controllers described in Section IV were implemented and run in addition to the VT&R software on a MacBook Pro with an Intel 2.4 Ghz Core2Duo processor and 4 GB of RAM. The input camera in both experiments was a Point Grey Bumblebee XB3 stereo camera. The resulting real-time localization and path-tracking control signals were generated at approximately 10 Hz. Since GPS was

not available, the improvement due to the ILC feedforward control signal was quantified by the localization of the VT&R algorithm.

B. Dealing with Initial Conditions

Since the initial conditions of each trial were set by the conclusion of a previous trial, we could not guarantee identical initial conditions for each trial. As such, we followed the approach of Freeman et al. [17] and smoothly modified the desired path to create a zero-error initial condition for the first k_{ic} vertices of each trial. For trial j, this is:

$$\mathbf{z}'_d[k_i] = \begin{cases} \mathbf{z}_d[k_i] - \frac{(k_{ic}-k_i)}{(k_{ic})}(\mathbf{z}_d[k_i] - \mathbf{z}_j[k_0]), & k_i < k_{ic} \\ \mathbf{z}_d[k_i], & k_i >= k_{ic}. \end{cases} \quad (3)$$

C. Results

In the first experiment, the ILC algorithm successfully reduced the maximum lateral and heading errors by factors of roughly three after six trials (see Figures 9 and 10). An example of the resulting re-localization is shown in Figure 3(b). For demonstration purposes, γ_u was set to 1.0, encouraging quick learning. However, generally, the perceived rate of convergence was often delayed and influenced by the evolution of the environment in response to the robot's activity. For example, repeating the same path caused ruts to form and sand located on side-hills to shift thereby causing new unmodelled disturbances and delaying the reduction in path-tracking errors. In addition, it was also found that the topography of the environment around the path resulted in changing disturbances from trial to trial. For example, if the path were along the ridge of a hill, the disturbances caused by the side-slope decreased from trial to trial as the robot path approached the ridge of the hill. While the ILC algorithm inherently continued to adjust its feedforward signal, in practice, γ_u should be set to less than 1.0 to give the system a greater ability to forget non-repeating disturbances and noise.

In the second experiment, the ILC algorithm successfully mitigated the effects of increased repeat speed over the course of ten trials. As can be seen in Figure 11, ILC was

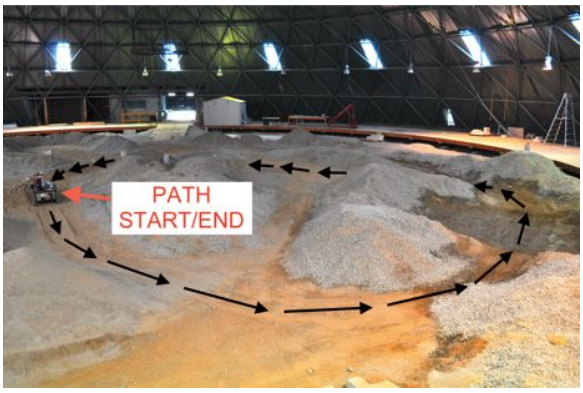


Fig. 8. Experiment 2: ROC6 at the start of the path in the MarsDome facility in Toronto, Ontario, Canada. The facility provides a network of sandy and gravel floor valleys defined by a set of impassable gravel hills. A supplemental video presenting this experiment is also available at http://ieeexplore.ieee.org.

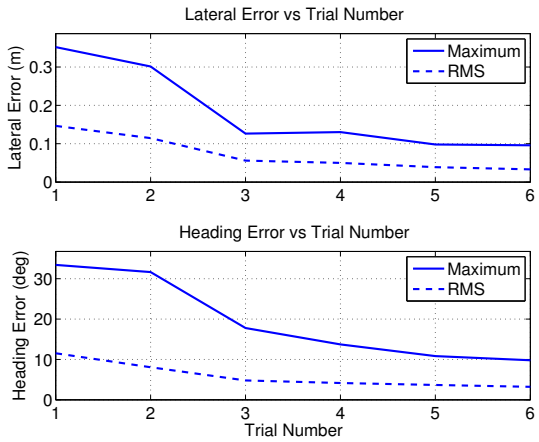
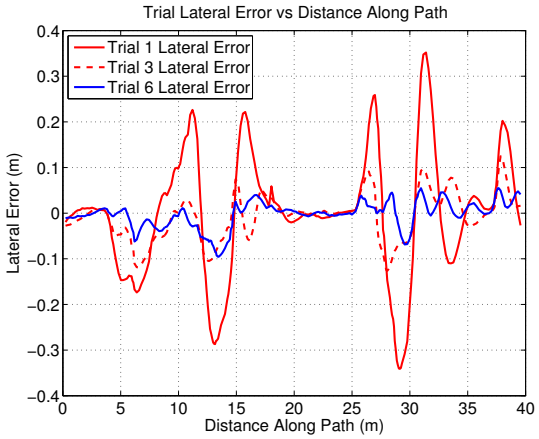
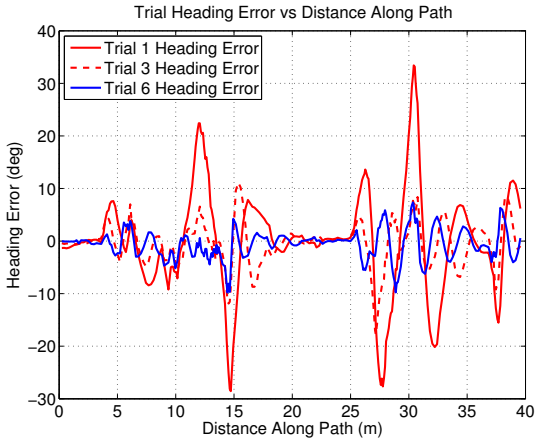


Fig. 9. Experiment 1 Results: Lateral and heading error versus trial number. The maximum and root mean square (RMS) errors are reduced significantly within the first few iterations.



(a) Experiment 1 Results: Lateral error versus distance along path.



(b) Experiment 1 Results: Heading error versus distance along path.

Fig. 10. The P-type ILC law was able to quickly reduce the maximum lateral and heading errors but had a greater challenge eliminating the errors completely.

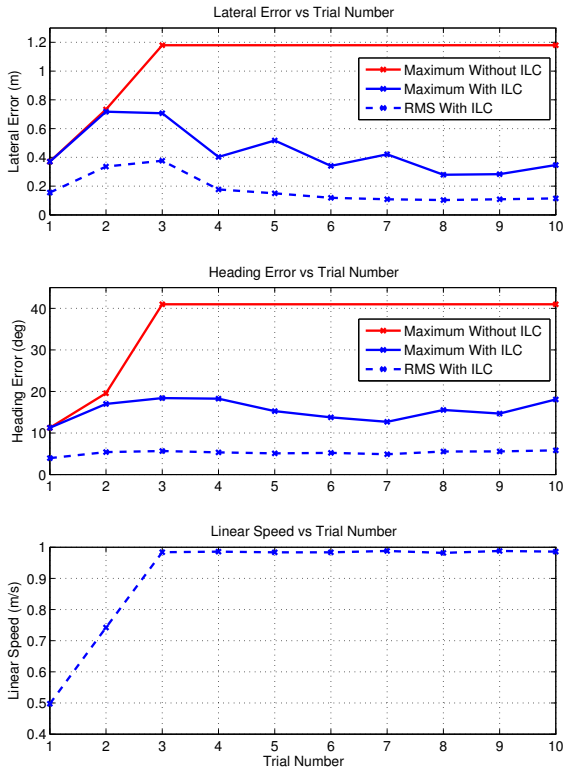


Fig. 11. Even though the ROC6 mass was 110 kg greater than that of the Husky A200, using the same ILC tuning parameters as experiment 1 resulted in decent performance in experiment 2. The given maximum heading and lateral errors for travel at 1.0 m/s without ILC are the values measured just prior to manual intervention.

used to gradually learn to drive at 1.0 m/s by sequentially increasing the repeat speed each trial. After ten trials, the ILC algorithm had reduced the lateral and heading errors by factors of roughly three compared to the errors when driving at 1.0 m/s without the ILC algorithm. Furthermore, when driving at 1.0 m/s without learning, two manual interventions were required to bring the robot back on the path from precarious locations on the sides of the valleys. Of note, without any other algorithm development, the process of learning to drive at a new speed implies forgetting the old. Interestingly, γ_u, γ_L, and γ_H were identical to those for the Husky suggesting the ILC update law has some independence of platform and path.

Overall, the choice to produce a spatially-indexed (as opposed to time-indexed) feedforward signal was largely made to conform to the vertex-based VT&R system and to facilitate changes in robot speed. The result was computationally simple and effective at reducing path-tracking errors.

VI. Conclusion

In summary, this paper presents an added-benefit, proportional-type Iterative Learning Controller for a path-repeating, mobile robot negotiating GPS-denied, extreme environments. The spatially-indexed, feedforward ILC operates in parallel to a feedback-linearization controller. Both controllers use on-board localization estimates relative to path vertices created and detected by an autonomous Visual Teach and Repeat system.

Two experiments, including over 600 m of travel, demonstrated the system's ability to handle unmodelled terrain and rover dynamics. During the first experiment, a four-wheeled 50 kg robot was taught a 40-m-long path including a 5-15° sandy side-slope. The ILC signal effectively reduced the lateral and heading errors by factors of three in the first six trials. During the second experiment, a six-wheeled 160 kg robot was taught a 50-m-long path with little margin for error across terrain with varying ground properties. ILC was used to gradually learn to drive at a speed three times the nominal repeat speed of 0.35 m/s. Compared to driving at 1.0 m/s without learning, the resulting lateral and heading path-tracking errors were reduced by a factor of three after ten trials.

References

[1] T. Barfoot, B. Stenning, P. Furgale, and C. McManus. Exploiting reusable paths in mobile robotics: Benefits and challenges for long-term autonomy. *In Proceedings of the 9th Canadian Conference on Computer and Robot Vision (CRV)*, pages 388–395, 2012.

[2] D. Nguyen-Tuong and J. Peters. Model learning for robot control: a survey. *Cognitive Processing*, 12:319–340, 2011.

[3] P. Furgale and T. Barfoot. Visual teach and repeat for long-range rover localization. *Journal of Field Robotics*, 27(5):534–560, 2010.

[4] S. Arimoto, S. Kawamura, and F. Miyazaki. Bettering operation of robots by learning. *Journal of Robotic Systems*, 1(2):123–140, 1984.

[5] D. Bristow, M. Tharayil, and A. G. Alleyne. A survey of iterative learning control. *IEEE Control Systems Magazine*, 26(3):96–114, 2006.

[6] H.-S. Ahn, Y. Q. Chen, and K. L. Moore. Iterative learning control: brief survey and categorization. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, 37(6):1099–1121, 2007.

[7] G. Oriolo, S. Panzieri, and G. Ulivi. An iterative learning controller for nonholonomic mobile robots. *International Journal of Robotics Research*, 17(9):954–970, 1998.

[8] M. Kang, J. Lee, and K. Han. Kinematic path-tracking of mobile robot using iterative learning control. *Journal of Robotic Systems*, 22(2):111–121, 2005.

[9] K-L. Han and J. Lee. Iterative path tracking of an omni-directional mobile robot. *Advanced Robotics*, 25:1817–1838, 2011.

[10] Y. Chen and K. Moore. A practical iterative learning path-following control of an omni-directional vehicle. *Asian Journal of Control*, 4(1): 90–98, 2002.

[11] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the Association for Computing Machinery*, 24(6):381–395, 1981.

[12] S. Segvic, A. Remazeilles, A. Diosi, and F. Chaumette. A mapping and localization framework for scalable appearance-based navigation. *Computer Vision Image Understanding*, 113(2):172–187, 2009.

[13] T. Krajnik, J. Faigl, V. Vonasek, K. Kosnar, M. Kulich, and L. Preucil. Simple yet stable bearing-only navigation. *Journal of Field Robotics*, 27(5):511–533, 2010.

[14] J. Marshall, T. Barfoot, and J. Larsson. Autonomous underground tramming for center-articulated vehicles. *Journal of Field Robotics*, 25(6-7):400–421, 2008.

[15] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision*, 74(3):237–260, 2007.

[16] C. Samson and K. Ait-Abderrahim. Feedback control of a nonholonomic wheeled cart in cartesian space. *In Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, 2:1136–1141, 1991.

[17] C. Freeman, Z. Cai, E. Rogers, and P. Lewin. Iterative learning control for multiple point-to-point tracking application. *IEEE Transactions on Control Systems Technology*, 19(3):590–600, 2011.