In *The IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS 07), San Diego, USA, October 2007.

Global Action Selection for Illumination Invariant Color Modeling

Mohan Sridharan
Electrical and Computer Engineering
The University of Texas at Austin, USA
smohan@ece.utexas.edu

Peter Stone
Department of Computer Sciences
The University of Texas at Austin, USA
pstone@cs.utexas.edu

***Abstract*— A major challenge in the path of widespread use of mobile robots is the ability to function autonomously, *learning* useful features from the environment and using them to *adapt* to environmental changes. We propose an algorithm for mobile robots equipped with color cameras that allows for smooth operation under illumination changes. The robot uses image statistics and the environmental *structure* to autonomously detect and adapt to both major and minor illumination changes. Furthermore, the robot autonomously plans an action sequence that maximizes color learning opportunities while minimizing localization errors. Our approach is fully implemented and tested on the Sony AIBO robots.**

***Index Terms*— Illumination invariance, Action sequence learning, Color segmentation, Legged robots.**

I. Motivation

Mobile robots are being used in fields as diverse as medicine, surveillance, rescue, and navigation [1]–[3] due to the availability of high-performance sensors such as laser and color cameras. Though color cameras are a rich source of sensory information, mobile robot applications have predominantly used non-visual sensors because cameras require manual calibration that needs to be repeated with environmental changes. In order to make the widespread use of color cameras more feasible, the robot needs to calibrate itself and adapt to changes. We focus on the specific task of *color segmentation*, i.e. the mapping from image pixels to color labels. Sophisticated vision algorithms [4], [5] are computationally expensive to implement on mobile robots that require real-time operation under constrained resources.

In the context of robots equipped with color cameras, this paper tackles two challenges to autonomous behavior: the sensitivity to illumination, and the need for color calibration. It makes two contributions:

1. The robot autonomously detects and adapts to illumination changes using environmental *structure* and autonomously-collected image statistics. Minor illumination changes are handled by *merging* new observations with existing models, and new models are learned in response to major illumination changes. Prior work has focused on tracking minor illumination changes [6] or detecting major ones [7]. We show that it is essential to adapt to *both* types of changes.
2. The robot autonomously generates an action sequence maximizing color learning opportunities while minimizing localization errors. Color information is needed for localization, which in turn is required to reliably reach locations suitable for learning colors. The robot learns a model to predict pose errors as a function of desired motion commands, and a statistical model on the feasibility of learning colors at various poses. The models are used to search for the best sequence of motion commands. Prior work has required labeled training samples [8] or a human-generated heuristic action sequence [7] for color learning.

All algorithms are implemented and tested on a physical robot. We show that the robot is able to autonomously learn colors and operate over a range of illuminations.

II. Related Work

Segmentation and color constancy are well-researched fields in computer vision. Several effective algorithms have been introduced [4], [5], [9], [10], including methods for learning colors and making segmentation robust to illumination changes [11], [12]. But most of the approaches are computationally expensive to implement on mobile robots with constrained resources.

On mobile robots the mapping from the pixels to color labels is typically created by hand-labeling image regions over a couple of hours [8]. In an attempt to learn this mapping, Cameron and Barnes [13] construct closed figures corresponding to known environmental features. The color information from these regions is used to build classifiers, but the approach requires human supervision and is time consuming even with the use of offline processing. Jungel [14] maintained layers of color maps with increasing precision levels, colors being represented as cuboids. But the segmentation is not as accurate as the hand-labeled one and is not robust to illumination changes. Schulz and Fox [15] estimate colors using a hierarchical Bayesian model with Gaussian priors and a joint posterior on robot position and illumination. The approach requires extensive prior information even for testing under two illuminations. Anzani et al. [16] model colors using a mixture of Gaussians and compensate for minor illumination changes by modifying the parameters. But prior knowledge of color distributions and suitable initialization of parameters are required. Thrun et al. [3] distinguish between two safe and unsafe road regions, modeling colors as a mixture of Gaussians whose parameters are updated using EM. Teams in the DARPA Learning Applied to Ground Robotics challenge detect safe regions using 3D color histograms [17]. These approaches to detecting safe regions do not help distinguish between overlapping colors. In recent work, Our prior work [7] presented a scheme to learn colors and detect large illumination changes. But it does not adapt smoothly to illumination changes, and learns colors using an action sequence based on human-specified heuristic functions. We extend the algorithm (and other methods [6]) to enable a robot to generate motion sequences most suitable for color learning, and to adapt smoothly to illumination changes.

III. Experimental Platform

The experiments reported here are run on the SONY *ERS-7* Aibo, a four-legged robot whose primary sensor is a CMOS camera at the tip of its nose, with a limited field-of-view (56.9^o horz., 45.2^o vert.). The images captured at 30Hz with a resolution of 208×160 pixels possess defects such as noise and distortion. The robot has 20 degrees-of-freedom, three in each leg, three in its head, and a five in its tail, mouth, and ears. It has wireless LAN for inter-robot communication. The legged motion results in jerky camera motion. All processing for vision, localization, motion and strategy is performed on-board using a 576MHz processor.

Fig. 1: An image of the Aibo and the field.

One major application domain for the Aibos is the RoboCup Legged League [18], a research initiative where teams of four robots play a competitive game of soccer on an indoor field ($4m \times 6m$). Applications on mobile robots with cameras typically involve an initial color calibration phase that needs to be repeated when illumination changes. Our approach enables the robot to autonomously generate a motion sequence to learn colors, and to adapt to illumination changes. Figure 1 shows the Aibo and the robot soccer environment. The robot uses the distances and angles to the detected objects to localize itself, using particle filtering.

IV. Problem Description

In order to operate in a color coded environment, the robot needs to recognize a discrete number of colors (N). A *color map* provides a color label for each point in the color space:

$$\Pi_E : \{m_{1,i}, m_{2,j}, m_{3,k}\} \mapsto l\,|_{l\in[0,N-1]}, \qquad (1)$$
$$\forall i,j,k \in [0,255]$$

where m_1, m_2, m_3 are the values along the three color channels (e.g. R, G, B), l refers to the numerical indices of the color labels (blue, orange etc), and E represents the dependence of the color map on illumination. Typically a human observer labels specific image regions (for $\approx$ 30 images) over a period of an hour or more and the color map is obtained by generalizing from the labeled samples [8]. The goals of this work are: (a) to autonomously generate a motion sequence that maximizes color learning opportunities while minimizing localization errors, and (b) to detect and adapt smoothly to a range of illuminations.

V. Algorithm

We present two novel algorithms: (a) the overall algorithm that learns colors, and detects and adapts to illumination changes (Algorithm 1), and (b) the algorithm that learns models and determines the *best* motion sequence for color learning (Algorithm 2). Specific details are described below.

Algorithm 1 Illumination Adaptation Algorithm.

```
Require: For each known illumination $E_i, i \in [0, M-1]$, color map $\Pi_{E_i}$, $(r,g)$ distributions $rgHist_{E_i}$, and distribution of JS-distances $D_{E_i}$.
Require: Algorithm to plan motion (Algorithm 2) and learn colors autonomously.
Require: Positions, shapes and color labels of the objects of interest in the robot's environment. Initial robot pose.
1: Initialize: $M = 0$, $illum = 0$, $testTime = 0$ (no prior illumination knowledge).
2: Plan motion and learn $\Pi_{E_{illum}}$.
3: Generate $rgHist_{E_{illum}}$, $N$ $(r,g)$ space distributions, and distribution of JS-distances, $D_{E_{illum}}$, using images captured at random during color learning.
4: Save image statistics, $M = M + 1$.
5: while true do
6:    Get new image. Segment image and detect objects.
7:    if $\underline{\text{minorChange}}(\ Color\ )$ then
8:       $\underline{\text{minorUpdate}}(\ Color\ )$. Then get $\Pi_{\widehat{E}}$ from current color distributions.
9:       Revise current illumination representation to get $rgHist_{\widehat{E}}$ and $D_{\widehat{E}}$, to be used for subsequent operations.
10:   end if
11:   if $currentTime - testTime \geq time_{th}$ then
12:      $rg_{test} = (r,g)$ distribution of current image.
13:      for $i = 0$ to $M - 1$ do
14:         $dAvg[i] = \frac{1}{N}\sum_j \underline{JSDist}(rg_{test}, rgHist_{E_i}[j])$
15:      end for
16:      if Exists( $\widehat{E}$ ) then
17:         $dAvg_{\widehat{E}} = \frac{1}{N}\sum_j \underline{JSDist}(rg_{test}, rgHist_{\widehat{E}}[j])$
18:      end if
19:      if Exists($\widehat{E}$) and $\underline{\text{withinRange}}(dAvg_{\widehat{E}}, D_{\widehat{E}})$ then
20:         Continue with $\Pi_{\widehat{E}}$.
21:      else if $\underline{\text{withinRange}}(dAvg[illum], D_{E_{illum}})$ then
22:         Continue with $\Pi_{E_{illum}}$.
23:      else if $\underline{\text{withinRange}}(dAvg[i], D_{E_i})$, $i \neq illum$ then
24:         Use $\Pi_{E_i}$, $illum = i$.
25:      else
26:         New illumination, $illum = M$, $M = M + 1$.
27:         Learn $\Pi_{E_{illum}}$ autonomously.
28:         Learn $rgHist_{E_{illum}}$ for new illumination.
29:         Use $\Pi_{E_{illum}}$ for subsequent operations.
30:      end if
31:      $testTime = currentTime$.
32:   end if
33: end while
```

A. Illumination Adaptation

The robot initially has no prior information on colors or illumination. It knows the positions, size and color labels of objects in its environment (structure), and its starting pose.

The robot uses the structure to plan a motion sequence and learn a representation for the color distributions of interest (Algorithm 2), which are used to generate the color map under the current illumination ($\Pi_{E_{illum}}$ – line 2).

1) Color and Illumination Representation: Patterned after prior work [7], we use a *disjunctive* representation that models the a priori probability density function (pdf) for each color l *either as a 3D Gaussian, or as a 3D Histogram*:

$$p(\mathbf{m}|l) \sim N(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l), \quad or \quad \equiv \frac{hist_l(b_1, b_2, b_3)}{\sum hist_l} \tag{2}$$

where (b_1, b_2, b_3) are the histogram bin indices corresponding to the color channel values $\mathbf{m} = (m_1, m_2, m_3)$. The histogram is normalized to obtain a pdf. Assuming all colors are equally likely, i.e. $P(l) = 1/N, \ \forall l \in [0, N-1]$, each color's *a posteriori* pdf is proportional to the a priori pdfs. The color space is discretized and each cell in the color map is assigned the label of the *most likely* color pdf. Depending on the test condition, one of the two models may be more suitable for a color's pdf, and the choice is made autonomously using the bootstrap test [19]. Though other color models are feasible, the disjunctive model provides a balance between accuracy and computation.

Each illumination is represented by a color map and autonomously-collected image statistics (line 3). Based on the hypothesis that images from the same illumination have measurably similar distributions of pixels in color space, images captured by the robot are transformed into the normalized RGB space (r, g, b), and the first statistic is a set of pdfs ($rgHist_{E_{illum}}$), each pdf being a color histogram in (r, g) with 64 bins in each dimension. The distance between every pair of pdfs is computed and the distribution of distances ($D_{E_{illum}}$), modeled as a Gaussian, constitutes the second statistic. The Jensen-Shannon (JS) measure is used for computing distance between distributions:

$$JS(\boldsymbol{a}, \boldsymbol{b}) = \frac{KL(\boldsymbol{a}, \boldsymbol{m}) + KL(\boldsymbol{b}, \boldsymbol{m})}{2} \tag{3}$$
$$KL(\boldsymbol{a}, \boldsymbol{b}) = \sum_i \sum_j (\boldsymbol{a}_{i,j} \cdot ln \frac{\boldsymbol{a}_{i,j}}{\boldsymbol{b}_{i,j}}), \ \boldsymbol{m} = \frac{\boldsymbol{a} + \boldsymbol{b}}{2}$$

The JS distance being a function of the log of the pdfs, is robust to peaks in the observed distributions, i.e. large image regions of a single color.

2) Adaptation To Changes: A mobile robot environment is subjected to a range of illuminations. Major illumination changes, for instance when the lamps are suddenly switched on/off, cause large shifts in color distributions. The current color map is no longer valid and the robot is soon lost. Minor/slow changes such as the variation in natural light during the day, cause the robot's segmentation performance to slowly deteriorate as the color distributions shift. We present a combined strategy for handling both these changes.

During normal operation the robot recognizes objects to localize and computes the following for each detected object:

$$\frac{numPixels_l}{totalPixels} \leq changeThreshold \tag{4}$$

where $numPixels_l$ represents the pixels of color l, the color label of the detected object, while $totalPixels$ is the total number of pixels within the object's bounding rectangle. If the ratio falls below a threshold consistently (more than 60% of N consecutive frames) it indicates a minor change in illumination, denoted by $Detect_{minor}$ (*minorChange()* – line 7). The new illumination is denoted by $\widehat{E}$ and the pixels within the corresponding image region are used to build a model and merge it with the current model, i.e. a histogram or a Gaussian (*minorUpdate()* – line 8). For Gaussians, we use the measurement update of a Kalman Filter [20]:

$$Gain\, \boldsymbol{K_l} = \boldsymbol{\Sigma}_{l_{old}} (\boldsymbol{\Sigma}_{l_{old}} + \boldsymbol{\Sigma}_{l_{new}})^{-1} \tag{5}$$
$$\boldsymbol{\mu}_{l_{up}} = \boldsymbol{\mu}_{l_{old}} + \boldsymbol{K_l}(\boldsymbol{\mu}_{l_{new}} - \boldsymbol{\mu}_{l_{old}})$$
$$\boldsymbol{\Sigma}_{l_{up}} = (\boldsymbol{I} - \boldsymbol{K_l})\boldsymbol{\Sigma}_{l_{old}}$$

where the subscripts old, new and up represent the current model, new observation and updated model respectively for color l. For histograms, a similar update scheme is used:

$$p_{l_{old}} = \frac{hist_{l_{old}}}{\sum hist_{l_{old}}}, \ p_{l_{new}} = \frac{hist_{l_{new}}}{\sum hist_{l_{new}}} \tag{6}$$
$$p_{l_{avg}} = w_{old} p_{l_{old}} + w_{new} p_{l_{new}}, \ w_{old} + w_{new} = 1$$
$$p_{l_{up}} = \frac{p_{l_{avg}}}{\sum p_{l_{avg}}}, \ hist_{up} = p_{l_{up}} \sum (hist_{l_{old}} + hist_{l_{new}})$$

The histograms are normalized to obtain pdfs, which are *merged* by weighted averaging (based on the ratio of the number of samples) and the updated histogram is determined. The color map and the current illumination model are modified and used in subsequent operations (line 9). This adaptation scheme is called $Adapt_{minor}$.

In order to detect sudden/large illumination changes, the robot periodically ($time_{th} = 0.5$sec) generates a test image histogram in the (r, g) space (line 12). The average distance ($dAvg$) is computed between this test histogram and the set of histograms corresponding to each illumination for which the robot has learned a representation (lines 13-15), using the JS distance – line 14. Any illumination representation obtained by tracking minor illumination changes is included in this computation (lines 16-18).

If $dAvg$ lies within the threshold range (95%) of the distance distribution corresponding to the current illumination (*withinRange()* – lines 19, 21), the robot continues to use the current color map. If $dAvg$ lies outside the range of the distance distribution of the current illumination, but within the range of the distance distribution corresponding to an illumination for which the robot has learned a model, the robot transitions to using the corresponding color and illumination models. But if $dAvg$ lies outside the range of all known illuminations, the robot detects a new illumination ($Detect_{major}$) and learns models for color and illumination (lines 25-30). If this adaptation scheme ($Adapt_{major}$) is used with a reduced threshold to handle minor illumination changes, it would result in a large number of color maps for minor changes in a few distributions. We will show experimentally that *both* $Adapt_{minor}$ and $Adapt_{major}$ are necessary for smooth operation under illumination changes.

B. Planned Color Learning

Our motion planning algorithm is described in Algorithm 2. In previous work [7], the known positions, size and color labels of the objects along a sequence of heuristically generated poses were used to extract image regions to be used as labeled samples. We enable the robot to determine a motion sequence that maximizes color learning opportunities while minimizing localization errors – the robot may obtain more training samples by moving a larger distance, but this motion may cause larger localization errors. We introduce three components: a motion error model, a statistical feasibility model, and a search routine.

```
Algorithm 2 Motion Sequence Generation.
Require: Ability to learn color models [7].
Require: Positions, shapes and color labels of the objects
    of interest in the robot's environment. Initial robot pose.
1: Move between randomly selected target poses.
2: CollectMEMData() – collect data for motion error model.
3: CollectColLearnStats() – collect color learning statistics.
4: NNetTrain() – Train the Neural network for the MEM, Equation 7.
5: UpdateFM() – Generate the statistical feasibility model, Equation 8.
6: GenCandidateSeq() – Generate candidate sequences, Equation 9.
7: EvalCandidateSeq() – Evaluate candidate sequences.
8: SelectMotionSeq() – Select final motion sequence.
9: Execute motion sequence and learn colors – Algorithm described in [7].
```

1) Motion Error Model (MEM): The MEM predicts the error in the robot pose (position and orientation) in response to a motion command, target (x, y, θ), as a function of the colors used for localization (the locations of color-coded markers are known). Assuming an even distribution of objects in the environment, the inputs are the difference between the starting pose and target pose, and the list of colors the robot has learned. The output is the pose error that would be incurred during this motion. The MEM is represented as a back-propagation neural network [21] with $N+3$ inputs, three outputs and one hidden layer of 15 nodes:

$$\{\Delta_x, \Delta_y, \Delta_\theta, c_1, c_2, \dots, c_N\} \mapsto \{err_x, err_y, err_\theta\} \quad (7)$$

where $\{\Delta_x, \Delta_y, \Delta_\theta\}$ represent the desired difference in pose, and $\{c_1, c_2, \dots, c_N\}$ are binary variables that represent the colors in the environment. If the robot knows all the colors it can recognize all the markers and localize well. With only some colors known, some markers aren't recognizable and localization suffers. During training the robot moves between poses running two localization routines, one with all colors known (provides ground truth) and another with only a subset of colors known. The difference in the two pose estimates provides the outputs for the training samples.

2) Statistical Feasibility Model (FM): For each robot pose, the FM provides the probability of learning each of the desired colors given that a certain set of colors have been learned previously. The possible robot poses are discretized into cells. Since the robot's joint angles and camera field-of-view are known, a feasibility check is performed to eliminate a lot of cells – if the robot's camera is not pointing towards any known object it cannot learn colors. This computation is performed once for each object configuration. Each cell of the FM also stores a probability measure:

$$FM(d, e, f, v_i) = p, \ \forall \{d, e, f\} \in [0, K-1] \quad (8)$$

where d, e, f are cell indices corresponding to the K discrete poses (x, y, θ), and $v_i, i \in [0, M-1]$ represents all possible combinations of colors. As the robot moves during training, its pose maps into one of the cells. Assuming prior knowledge of a set of colors, it attempts to learn other colors and stores a count of successes. At the end of the training phase Gaussian-smoothing is used to smooth out the noise, and the normalized cell counts provide the probability.

3) Search for Motion Sequence: In the training phase, the robot moves between randomly generated target poses and collects the data to build the MEM (*CollectMEMData()* – line 2) and the statistics for the FM (*CollectColLearnStats()* – line 3). The FM has to be re-learned when the object configurations change, but even with just the geometric constraints the robot is able to provide motion sequences leading to successful color learning. The robot uses the collected data to estimate the parameters of MEM (*NNetTrain()* – line 4) and the probability values in the FM (*UpdateFM()* – line 5). Then it iterates through all candidate motion sequences (*GenCandidateSeq()* – line 6), i.e. all possible paths along the discretized pose cells. The depth of the search is equal to the number of colors to be learned.[1] If the robot is to learn N colors, the motion sequence is a path:

$$path : \{x_i, y_i, \theta_i, color_i\} \quad \forall i \in [0, N-1] \quad (9)$$

This formulation results in a large number of paths ($\approx 10^9$). But only a smaller subset of paths ($\approx 10^4$) are evaluated completely. The MEM provides the expected pose error if the robot travels from the starting pose to the first pose. The vector sum of the error and the target pose provides the actual pose. If the desired color can be learned at this pose, the move to the next pose in the path is evaluated. If the whole path is evaluated, the net pose error and probability of success are computed (*EvalCandidateSeq()* – line 7). Of the paths that provide a high probability of success, the one with the least pose error is used by the robot (*SelectMotionSeq()* – line 8) to learn the colors [7].

VI. Experimental Results

We need to test the robot's ability to: (a) plan a motion sequence and learn a color map for different configurations, and (b) use the color map and image statistics to detect and adapt to a range of illuminations.

A. Motion Planning Experiments

Segmentation accuracy is not a good performance measure in the presence of background noise. Instead, we measure the localization accuracy. Of the colors needed for localization (*pink, yellow, blue, white, green*), the ground

[1]We assume that the robot learns one color at each pose.

colors (*green*, *white*) are learned by scanning in place, i.e. the depth of the search process is three. The range of poses was divided into $(6 \times 9 \times 12)$ cells, i.e. divisions of 600mm, 600mm, and 30^o along x, y, amd θ. The back-propagation network was learned using the MATLAB Neural Network toolbox [22] ($\leq$ 1 min for $\approx$ 2000 training samples). The final search process takes ≤ 4 mins.

We tested the planning under different object configurations – there are six objects that can be placed anywhere along the outside of the field, but the robot knows their positions. As described in [7], we asked a group of graduate students to suggest challenging configurations and robot starting poses. The planning accuracy averaged over 5 different object configurations, each with 15 different robot starting poses, is shown in Table I. We also had the robot move through a set of poses using the learned color map and measured localization errors (15 trials of 10 poses) – a tape measure and protractor provided ground truth.

Config	Plan (%)	Localization error		
		X (cm)	Y (cm)	θ (deg)
Learned	100	9.6 ± 3.7	11.1 ± 4.8	9 ± 7.7
Hand-labeled	–	6.9 ± 4.1	9.2 ± 5.3	7.1 ± 5.9

TABLE I: Planning and Localization Accuracies in challenging configurations. Planned motion sequence always succeeds in learning colors. Localization comparable to hand-labeled color map.

The robot is able to generate a valid plan over *all* the trials, unlike human-generated heuristic planning (90% success [7]). The localization accuracy is comparable to that obtained from a hand-labeled color map.

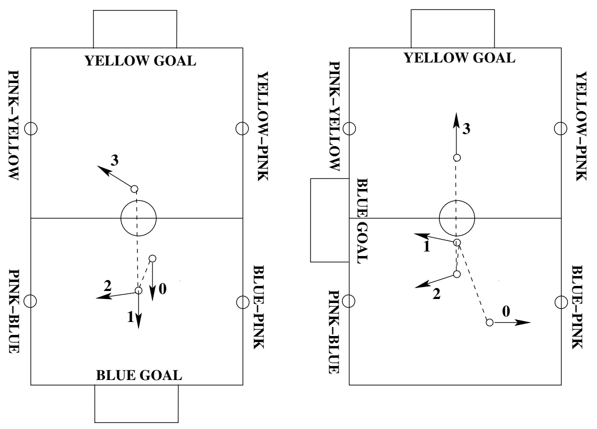

Fig. 2: Sample motion plans generated by the algorithm. All plans lead to successful color learning on the robot.

Figure 2 shows some planning results, where the starting position is denoted by number '0' while the direction of the arrows show the orientation. We observe the following:

1. The robot never attempts to learn the color *pink* first. Since all objects with the color *pink* have another colored blob of the same size, it is better to learn *pink* after one of the other two colors (*blue, yellow*) have been learned (see field description [23]).
2. Among the other two colors (*blue, yellow*) the robot first learns the color which requires the execution of a motion command that is least likely to cause error.
3. For colors which exist in several objects the robot makes a trade-off between a larger object that provides more training samples but is farther, and a smaller object that is closer.

In addition to the 'best' motion-plan, several of the top sequences lead to successful color learning. Over a set of 20 images, the average segmentation accuracy of the learned and hand-labeled color map are 94.9 ± 3.9 and 96.7 ± 4.3 respectively (no difference at 95% significance). Ground truth is provided by a human observer. The color learning and motion planning takes around 7 minutes in comparison to the heuristic motion planning or hand-labeling that takes more than an hour of human effort. The motion planning is particularly useful where object configurations change less frequently than illumination. Sample planning results, images and a video of the learning algorithm can be viewed at: *www.cs.utexas.edu/~AustinVilla/?p=research/autoplan_illum*

B. Illumination Adaptation Experiments

The main hypothesis to test is that it is essential to include schemes for both major and minor illumination changes. In these experiments, using $Adapt_X$ implies the use of $Detect_X$ as well.

To show that $Adapt_{minor}$ is needed, we let the robot use $Adapt_{major}$ but slowly changed the illumination (over 20secs) between two conditions that are not significantly different to be detected by $Detect_{major}$. The robot stood in place and moved its head from left to right, averaging the measured distance and angle to an object over the 20sec interval. Table II shows the errors in measurements compared to the ground truth values obtained with a tape measure and protractor, both with and without $Adapt_{minor}$. Results are averaged over four different objects and three different illumination sources (different color temperatures) with ≈ 15 trials under each case.

Illum + Alg	Dist error (mm)	Ang error (deg)
$Slow + NoAdapt$	191.31 ± 105.61	12.37 ± 2.85
$Slow + Adapt_{min}$	25.53 ± 19.14	2.11 ± 0.83

TABLE II: Error in distance measurements with and without $Adapt_{minor}$. Adaptation results in much smaller errors.

In the absence of $Adapt_{minor}$, as the illumination is slowly changed, the segmentation performance slowly deteriorates until the object is no longer recognizable. There are instances where the robot detects an object incorrectly, leading to the errors in distance and angle measurements, and hence localization errors. Using $Adapt_{minor}$ leads to segmentation accuracy (95.1 ± 4.3), and hence localization errors ($\approx$ 10cm, 12cm and 10deg) similar to those under constant illumination.

The results show that using $Adapt_{major}$ without $Adapt_{minor}$ can lead to segmentation and localization errors, affecting the robot's performance. Next, to show that $Adapt_{major}$ is essential we had the robot *find-and-walk-to-object* and measured the time taken to perform the task.

The robot started out near the center of the field with the object placed near the penalty box of the opponent's goal. Table III tabulates the results under six different cases, averaged over three illumination sources, with 15 trials in each test condition.

Illum + Alg	Time (sec)	Fail
$Constant + NoAdapt$	6.18 ± 0.24	0
$Slow + Adapt_{maj}$	31.73 ± 13.88	9
$Slow + Adapt_{maj,min}$	6.24 ± 0.31	0
$Sudden + Adapt_{min}$	45.11 ± 11.13	13
$Sudden + Adapt_{maj,min}$	9.72 ± 0.51	0
$Sudden + Slow + Adapt_{maj,min}$	10.32 ± 0.83	0

TABLE III: Time taken to find-and-walk-to-object.

When the illumination does not change, the robot can *find-and-walk-to-object* in 6.18 ± 0.24 seconds. When the illumination is slowly changed as the robot performs the task, using just $Adapt_{major}$ does not help – large variance in second row. With $Adapt_{minor}$ the results are as good as before (6.24 ± 0.31secs). Next, when the illumination is suddenly changed as the robot starts walking towards the seen object, using just $Adapt_{minor}$ does not help. The robot totally fails to perform the task most of the time, as seen by the large number of failures (fourth row, third column). With the combined strategy, i.e. with both $Adapt_{major}$ and $Adapt_{minor}$, the robot can perform the task efficiently, the additional time being used to ensure that a change in illumination did occur (9.72 ± 0.51secs). We infer that the improvement is primarily due to $Adapt_{major}$. In all these experiments, when the illumination changes significantly, the robot is put in conditions similar to the ones for which it has already learned color and illumination models. Once $Detect_{major}$ is triggered, $Adapt_{major}$ consists of transitioning to the suitable representation.

Finally the robot is made to *find-and-walk-to-object* while the illumination is changed significantly initially, and after 3sec is changed slowly over the next 5sec. The robot is able to do the task in 10.32 ± 0.83sec *iff* both $Adapt_{major}$ and $Adapt_{minor}$ are used. Therefore, a combination of the schemes (Algorithm 1) is essential to operate under a range of illumination intensities ($\approx 400Lux$ to $\approx 1600Lux$) and color temperatures ($2300K$ – $4000K$). Images and videos of the robot's performance in response to minor and major illumination changes are available online: *www.cs.utexas.edu/∼AustinVilla/?p=research/autoplan_illum*

VII. Conclusions and Future Work

Mobile robots are being used extensively in real-world applications. But their full potential can be exploited only if they function autonomously. For mobile robots equipped with color cameras, two major challenges are the manual calibration and the sensitivity to illumination. Prior work has managed to learn a few distinct colors [3], model known illuminations [10], and use heuristic action sequences to facilitate learning and detect sudden changes [7].

We propose an algorithm that enables a mobile robot to learn functions that maximize color learning opportunities while minimizing localization errors – robot learns color autonomously. Furthermore, the robot is able to detect and adapt to a range of illuminations. The resulting color segmentation and localization are comparable to that obtained by a hand-labeled color map. The algorithm requires the environmental structure as input, but the structure is much easier to provide than hand-labeling several images.

A future direction of research is to address motion-planning in a rigorous linear programming framework with suitable constraints. Another challenge is to combine this work with autonomous vision-based map building (SLAM) [24]. Ultimately we aim to develop efficient algorithms for autonomous mobile robot operation under completely uncontrolled natural conditions.

References

[1] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun, "Towards robotic assistants in nursing homes: Challenges and results," *RAS Special Issue on Socially Interactive Robots*, 2003.

[2] B. W. Minten, R. R. Murphy, J. Hyams, and M. Micire, "Low-order-complexity vision-based docking," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 922–930, 2001.

[3] S. Thrun, "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[4] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *PAMI*, 2002.

[5] J. Shi and J. Malik, "Normalized cuts and image segmentation," *In IEEE Transactions on PAMI*, 2000.

[6] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric Model for Background Subtraction," in *The European Conference on Computer Vision (ECCV)*, 2000.

[7] M. Sridharan and P. Stone, "Color learning on a mobile robot: Towards full autonomy under changing illumination," in *The International Joint Conference on Artificial Intelligence*, 2007.

[8] D. Cohen, Y. H. Ooi, P. Vernaza, and D. D. Lee, *UPenn TDP, RoboCup-2003: RoboCup Competitions and Conferences*, 2004.

[9] L. T. Maloney and B. A. Wandell, "Color Constancy: A Method for Recovering Surface Spectral Reflectance," *Journal of Optical Soceity of America A*, vol. 3, no. 1, pp. 29–33, 1986.

[10] C. Rosenberg, M. Hebert, and S. Thrun, "Color constancy using kl-divergence," in *The IEEE International Conference on Computer Vision (ICCV)*, 2001.

[11] Y. B. Lauziere, D. Gingras, and F. P. Ferrie, "Autonomous physics-based color learning under daylight," in *Conf. on Color Techniques and Polarization in Industrial Inspection*, 1999.

[12] T. Gevers and A. W. M. Smeulders, "Color based object recognition," *In Pattern Recognition*, 1999.

[13] D. Cameron and N. Barnes, "Knowledge-based autonomous dynamic color calibration," in *The International RoboCup Symposium*, 2003.

[14] M. Jungel, "Using layered color precision for a self-calibrating vision system," in *The International RoboCup Symposium*, 2004.

[15] D. Schulz and D. Fox, "Bayesian color estimation for adaptive vision-based robot localization," in *IROS*, 2004.

[16] F. Anzani, D. Bosisio, M. Matteucci, and D. Sorrenti, "On-line color calibration in non-stationary environments," in *The International RoboCup Symposium*, 2005.

[17] "The DARPA Learning Applied to Ground Robots Challenge," 2005, www.darpa.mil/ipto/programs/lagr/.

[18] H. Kitano, M. Asada, I. Noda, and H. Matsubara, "Robot world cup," *Robotics and Automation*, 1998.

[19] B. Efron and R. J. Tibshirani, *An Introduction to Bootstrap*. Chapman and Hall Publishers, 1993.

[20] P. S. Maybeck, *Stochastic Models, Estimation and Control*, 1979.

[21] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[22] "Matlab Neural Network Toolbox," http://www.mathworks.com/access/helpdesk/help/toolbox/nnet/.

[23] "The Four-Legged League," 2007, http://www.tzi.de/4legged/.

[24] P. Jensfelt, J. Folkesson, D. Kragic, and H. I. Christensen, "Exploiting distinguishable image features in robotic mapping and localization," in *The European Robotics Symposium (EUROS)*, 2006.