2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Optimality Framework for Hausdorff Tracking using Mutational Dynamics and Physical Programming*

Amit Goradia†, Clayton Haffner‡, Ning Xi† and Matt Mutka‡
†*Department of Electrical and Computer Engineering*, ‡*Department of Computer Science and Engineering*
Michigan State University
East Lansing, Michigan, 48824, USA
{*goradiaa, haffnerc, xin, mutka*}*@egr.msu.edu*

***Abstract*— The task of visual surveillance involves pervasively observing multiple targets as they move through a field of sensor nodes. Mutational analysis and shape based control have been proposed to overcome the limitations of current feature (point) based visual servoing and tracking techniques generally employed to provide an optimal solution for the surveillance task. Hausdorff tracking paradigm for visual tracking of multiple targets using a single sensor has been proposed for accomplishing the surveillance task. However, Hausdorff tracking incorporates some redundancy in the actuation mechanism. This paper exploits this redundancy in the camera motion in order to accomplish various sub-tasks which can be assigned to the system, such as minimization of consumed energy maintaining manipulability etc. The complete task can then be expressed in a multi-objective constrained optimization framework and can be solved, i.e., the input to the camera can be derived, using various methods such as physical programming, nonlinear programming, weighted sum method, etc. In this paper, we use the physical programming method based on the various advantages such as ease of expressing multiple objectives in a physically significant manner. Experimental results are provided which show the advantages of using the physical programming approach over the weighted sum method for constructing the task criterion for multi-objective optimization problems.**

I. Introduction

The task of a surveillance camera is to continuously keep a single or multiple moving targets in its active Field Of View (FOV). The surveillance task can be expressed, using shape functions [1], as the minimization of a Hausdorff distance-based metric or the size of the target, etc. The shape function represents the error between the desired and actual shapes and reducing it to zero will accomplish the task. In our earlier research [2], the method of Hausdorff tracking using mutational equations for performing the surveillance task was presented.

Due to the redundancy in choice of input when using the Hausdorff tracking method, there are numerous choices of the input applied to the camera which can accomplish the task. Hence we need to select an appropriate motion input to the camera for task accomplishment. We can select various sub-tasks that the sensor can perform in order to utilize the redundancy offered by the camera motion. For example, the energy consumed due to the motion of the sensor should also be minimized while maintaining a safe distance from sensed obstacles (in the case of sensors mounted on mobile robots). These combined tasks can be expressed as a multi-objective optimization problem which can then be solved using various multi-objective constrained optimizations techniques such as weighted sum method, physical programming, nonlinear programming, search space methods, etc.

Surveillance systems are designed to perform tracking tasks with different requirements under varying environmental conditions. Furthermore, a dynamically varying environment implies the task criterion used for optimization should be generated on-line.

This paper investigates the setup of an optimal control framework, using dynamic multiple objective optimization, for the Hausdorff tracking problem. Further, this paper uses physical programming to solve the optimization problem. The advantage of using physical programming over the weighted sum method is that physical programming attaches a physical meaning to the various criteria functions which makes the problem more intuitive to the designer.

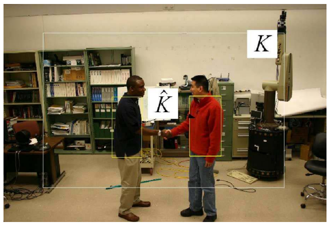
II. Hausdorff Tracking

Multiple target coverage can be readily expressed in a set based topological framework using shape analysis and shape functions [1], [3]. Thus, the variables to be taken into account are no longer vectors of parameters but the geometric shapes (domains) themselves. However, due to the lack of a vectorial structure of the space, classical differential calculus cannot be used to describe the dynamics and evolution of such domains. Mutational analysis endows a general metric space with a net of “directions” in order to extend the concept of differential equations to such geometric domains. Using mutational equations, we can describe the dynamics (change in shape) of the sensor FOV and target domains and further derive feedback control mechanisms to complete the specified task.

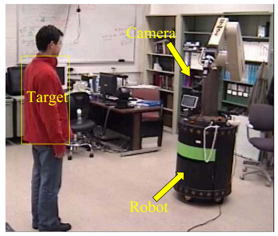
Shape or a geometric domain can be defined as the set $K \in \mathcal{K}(E),\ E \subset \mathbb{R}^n$ where $\mathcal{K}(E)$ represents the space of all non-empty, compact subsets of E. The target and the camera coverage can be readily expressed as shapes. Mutational equations can then be used to express the change (deformation) in the coverage and target sets based on the motion of the sensor. Shape analysis [1] can be used to address problems involving

* This research is supported in part by NSF Grants OCI-0334035 and CNS-0551464

1-4244-0602-1/07/$20.00 ©2007 IEEE.



(a) Target set $\hat{K}$ and coverage set K for image-based Hausdorff tracking.



(b) Target tracking using a Sony EVI-D30 camera mounted on Robot Nomad XR-4000 mobile robot

Fig. 1. Tracking scenario

geometric domains or shapes. Shape functions, which are set-defined-maps from $J(K) : \mathcal{K}(E) \mapsto \mathbb{R}$, can be used to provide a "measure" of acceptability and optimality of the shape K. For example, we can use a shape function to see if a reference set $\hat{K}$ is contained within a current set K. In order to accomplish the task defined using shape functions, we need to derive a feedback map $\mathcal{U} : \mathcal{K}(E) \mapsto U$, where $u = \mathcal{U}(K(t))$ is the input to the sensor, which will reduce the shape function to zero. The convergence of the shape function can be analyzed using the shape Lyapunov theorem [4]. The convergence to zero of the shape function for a particular task would imply task accomplishment.

1) Target, Coverage Sets and Shape Functions: The target set $\hat{K}$ is represented as the set of pixels comprising the target and the sensor coverage set is represented as a rectangle at the image center, K, as shown in Figure 1(a). The target set can be disjoint, which allows for multiple modeling the multiple target coverage task. For the multiple target coverage problem, the shape function is chosen as:

$$J(\hat{K}) = \int_{\hat{K}} d_K^2(q)\; dq \tag{1}$$

where, $q \in \hat{K}$ and $d_K(q) = \{\|q - p\| \; |p \in K, q \in \hat{K}\}$ is the directed demi-Hausdorff distance of the target set $\hat{K}$ from the coverage set K. Note that the shape function $J(\hat{K})$ is zero only when set $\hat{K}$ is completely covered by set K. Otherwise $J(\hat{K})$ is a non-zero positive value.

2) Dynamics Model Using Mutational Equations: Sets or domains evolving with time are called tubes and can be defined as a map $K(\cdot) : \mathbb{R}^+ \mapsto \mathcal{K}(E)$. The deformation (motion) of the coverage and the target sets can be represented using tubes. The evolution of a tube can be described using the notion of a time derivative of the tube as the perturbation of a set. Associate with any Lipschitz map $\varphi : E \mapsto E$, a map called the transition $\varrho_\varphi(h, q) := q(h)$, which denotes the value at time h of the solution of the differential equation: $\dot{q} = \varphi(q),\; q(0) = q_0$. Extend this concept of a transition to the space $\mathcal{K}(E)$ by introducing the reachable set from set K at time h of φ as

$$\vartheta_\varphi(h, K) := \{\varrho_\varphi(h, q_0)\}_{q_0 \in K} \tag{2}$$

For defining mutational equations, we supply the space $\mathcal{K}(\mathbb{R}^n)$ with a distance $d\!\!l$, for example the Hausdorff distance between domains $K_1, K_2 \in \mathcal{R}^n$ defined by $d\!\!l(K_1, K_2) = \sup_{q \in \mathbb{R}^n} \|d_{K_1}(q) - d_{K_2}(q)\|$, where, $d_K(q) = \inf_{p \in K} \|q - p\|$ represents the distance between the point q and set K.

Using the concept of the reachable set the time derivative of a tube can be defined as a mutation:

Definition 1: (Mutation) Let $E \subset \mathbb{R}^n$ and $\varphi : E \mapsto E$ be a Lipschitz map ($\varphi \in Lip(E, \mathbb{R}^n)$). If for $t \in \mathbb{R}^+$, the tube $K : \mathbb{R}^+ \mapsto \mathcal{K}(E)$ satisfies:

$$\lim_{h \to 0^+} \frac{d\!\!l(K(t+h), \vartheta_\varphi(h, K(t)))}{h} = 0, \tag{3}$$

then, φ is a mutation of K at time t and is denoted as:

$$\mathring{K}(t) \ni \varphi(t, K(t)),\;\; \forall t \geq 0 \tag{4}$$

It should be noted that φ is not a unique representation of the mutational equation. This justifies the use of the notation ($\ni$) [5].

We can further define controlled mutational equations as:

$$\mathring{K}(t) \ni \varphi(t, K(t), u(t)),\;\; \forall t \geq 0,\;\; u(t) \in U \tag{5}$$

A feedback law can be defined as a map $\mathcal{U} : \mathcal{K}(E) \mapsto U$ associating a control u with a domain $K(t)$ as: $u(t) = \mathcal{U}(K(t))$. Using a controlled mutational equation, we can model the motion of the target and coverage sets due to the motion input u to the camera as:

$$\mathring{K}(t) \ni \varphi(K, u) := \{\dot{q} = \varphi(q, u) | q \in K\} \tag{6}$$

The deformation of the target set due to the motion of the camera can be represented using a controlled mutational equation and can be modeled using optic flow equations as:

$$\begin{bmatrix} \dot{q}_x \\ \dot{q}_y \end{bmatrix} = \varphi(q, u) = B(q) \begin{bmatrix} u_c \\ \dot{\lambda} \end{bmatrix} = B(q)u \tag{7}$$

where, a point $P = [x, y, z]^T$, whose coordinates are expressed with respect to the camera coordinate frame projects on to the image plane with coordinates $q = [q_x, q_y]^T$ and λ is the focal length [6].

Using Equation (7) the mutational equation [2], [5] of the target set can be written as a collection of motion equations for the points comprising the set $\hat{K}$ as:

$$\begin{aligned} \dot{q} &= \varphi(q, u) = B(q)u \\ \mathring{\hat{K}} &\ni \varphi(\hat{K}, u) \end{aligned} \tag{8}$$

3) Feedback Map u: The problem now is to find a feedback map u such that the shape function J is reduced to zero. For this purpose, we need to find the shape directional derivative $\mathring{J}(\hat{K})(\varphi(\hat{K}, u))$ of $J(\hat{K})$ in the direction of the mutation $\varphi(\hat{K}, u)$ which represents the change in the shape function due to the deformation of the target set $\hat{K}$. From [5] and [3], the directional derivative of the shape function having the form of Equation (1) can be written as:

$$\mathring{J}(\hat{K})(\varphi(\hat{K}, u)) = \int_{\hat{K}} \nabla d_K^2(q)\varphi(q) + d_K^2(q)\text{div}\varphi(q)\; dq \tag{9}$$

The asymptotic behavior of the measure $J(K(t))$ of the deformation of the set K can be studied using the shape Lyapunov theorem [4], which provides the conditions to guarantee the convergence of $J(K(t))$ to 0.

Theorem 1: Consider $E \subset \mathbb{R}^n$ and a mutational map φ defined on the set E, a shape function $J : \mathcal{K}(E) \mapsto \mathbb{R}^+$ and a continuous map $f : \mathbb{R} \mapsto \mathbb{R}$. Let the Eulerian semi-derivative of J in the direction φ exist and be defined as $\mathring{J}(K)(\varphi(K,u))$. The function J is an f-Lyapunov function for φ if and only if, for any $K \in \mathrm{Dom}(J)$, we have

$$\mathring{J}(K)(\varphi(K,u)) + f(J(K)) \leqslant 0. \tag{10}$$

See [4] for proof.

Using the shape Lyapunov theorem, we can find the assumptions on input u such that the shape function $J(\hat{K})$ tends to zero as:

$$C(\hat{K})u \leqslant -\alpha J(\hat{K}) \tag{11}$$

where, $\alpha > 0$ is a scalar gain value such that the scalar system $\dot{w} = -\alpha w$ is stable in the sense of Lyapunov.

The feedback map u, which is an input to the camera module, can be calculated from (11) using the notion of a generalized pseudoinverse $C^{\#}(\hat{K})$ of the matrix $C(\hat{K})$ as:

$$u \leqslant -\alpha C^{\#}(\hat{K})J(\hat{K}) \tag{12}$$

It is important to note that the gain distribution between the various redundant control channels depends on the selection of the null space vector when calculating the generalized pseudoinverse $C^{\#}$ of matrix C.

A. Optimal Hausdorff Tracking

Given the task specified by Equation (11), notice that $C(K, \hat{K})$ is not a square matrix, which indicates that the system is redundant. This implies that the system has infinite solutions for a choice of input u which will accomplish the task. Therefore the designer can choose an algorithm in order to achieve an optimal solution for this system.

Optimality can be quantified by defining supplementary task functions, called objective functions, which the system needs to minimize to achieve an optimal solution. The optimization problem can be expressed as: Find the target and coverage sets (K and $\hat{K}$ respectively) and input u which satisfy:

$$\begin{array}{ll} \min\limits_{K,\hat{K},u} : & \mathbf{D}(K,\hat{K},u) = \left[\begin{array}{ccc} D_1(K,\hat{K},u) & \ldots & D_N(K,\hat{K},u) \end{array}\right]^T \\ \text{Subject to:} & \begin{cases} J(K,\hat{K}) = \int_{\hat{K}} d_K^2(p)\ dp \\ \mathring{J}(K,\hat{K})(\varphi_1,\varphi_2) = C(K,\hat{K})u \leqslant -\alpha J(K,\hat{K}) \end{cases} \end{array} \tag{13}$$

where, $D_i(K, \hat{K}, u),\ i = 1 \text{to} N$ are the N objective functions $D(K, \hat{K}, u) \in \mathbb{R}^N$ is a vector of these objective functions.

For the case of visual surveillance, the choice of objective functions can include (but is not restricted to):

1) Energy consumed which can be represented by $\|u\|^2$.
2) Resolution constraints which can be represented as $Area\_min \leqslant \int_{\hat{K}} dp \leqslant Area\_max$.

Various methods such as weighted sum approach and physical programming can be used to solve the above optimization problem. It is proposed to use physical programming for solving the constrained multi-objective optimization problem. The significant advantage of the physical programming approach is it capability of placing the design into a more flexible and natural framework.

III. Methodology of Optimal Task Distribution for Target Tracking

The dynamic optimal task distribution algorithm depicted in Figure 2 consists of three modules: task analyzer, optimization scheduling, and optimal solution calculation. The task analyzer takes current condition of motion module, target state and working environment as inputs, analyzes requirements of the current task, and generates a collection of parameters called as Task Indices (TI) to represent task requirements. The task analyzer passes the TI to the optimization scheduling module which then builds the task criterion function. The optimal solution calculation algorithm then finds an optimal solution for the criterion.

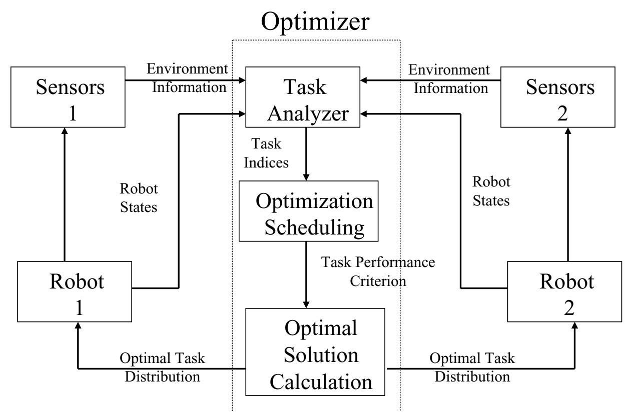


Fig. 2. The architecture of the optimal task distribution scheme.

A. Task Analyzer

The task analyzer takes as input the various states of the task including the location and states of the various sensors, targets and obstacles and provides a task index vector as an output which represents the requirements of the task in a concise form.

For optimal performance, the task criterion function should be dynamically modified according to time varying task requirements. An on-line task analyzer is proposed to analyze task requirements. Using the information from the sensors and the working environment as inputs, a collection of task indicators is generated online, which represent the actual conditions of the tracking task being performed. Using a fuzzy inference scheme, the task analyzer maps the task conditions represented by the task indicator vector $\mathbf{u} \in R^M$, into TI, which represents task requirements.

The Task Indices (TI) vector is used to mathematically describe the task requirements. These indices are generated by a fuzzy mapping which takes the task indicators as input variables. Various task requirements have their own physical and functional meaning according to which they can be categorized into classes. For example various classes such as maintaining the manipulability of the motion module, singularity avoidance, conserving energy, maintaining adequate target resolution, etc., can be used.

For each class of task requirement, we generate one Task Index. For all task requirements, a vector of Task Indices is generated:

$$\begin{aligned} \textbf{Index} &= \{Index_1, Index_2, \cdots, Index_K\}^T, \\ Index_i &= f_i(\mathbf{u}) = \frac{\sum_{j=1}^{M} a_{i,j} g_{i,j}(u_j)}{\sum_{j=1}^{M} a_{i,j}}. \end{aligned} \quad (14)$$

where, $Index_i$ is the indicator analyzer for the i^{th} requirement class. $a_{i,j}$ is the relative weight of task indicator u_j in the task requirement class i, $g_{i,j}$ is the mapping function of the task indicator u_j for the task requirement class i. (14) maps the task indicator value into the extent of task requirement. The relative weights are assigned based on the physical meaning and importance of a particular task indicator within a particular task requirement class.

B. Optimization Scheduling

This module generates the task criterion based on task requirements represented in the TI using physical programming. This module consists of two parts: physical programming and PP (Physical Programming) analyzer.

Physical programming, which is a multi-objective optimization approach, maps the collection of objective functions into a utility function. Since the task requirements vary according to transformation of tasks and environment conditions, the mapping model for physical programming should change with the task requirements. Therefore, the PP analyzer is built to analyze the physical programming mapping model based on the TI. A confidence vector $\Xi \in R^N$,

$$\Xi = \{\xi_1, \xi_2, \cdots, \xi_N\}^T \quad (15)$$

is generated to represent the degree of the system designer's satisfaction about physical programming mapping model according to the current task's conditions. Finally, physical programming mapping is modified based on Ξ.

1) Physical Programming: Optimal task distribution for Hausdorff tracking using multiple robots is a multi-objective optimization problem for optimizing a vector of objective functions, **D** in Equation (13). Usually, the system designer maps the collection of multiple objective functions C_i into a utility function $Z \in R$, which we call the task criterion, using a mapping $P : R^N \rightarrow R$ as:

$$Z = P(D_1, D_2, \cdots, D_N) \quad (16)$$

D_i is the objective function for the i^{th} task requirement. The optimal solution is achieved by choosing the system variables $[u, p]$ which endows the optimal value for the criterion Z, where u is the input velocity of the various axes and p is the state of the sensors and targets.

2) Weighted Sum Method: The weighted sum method is the most popular mapping method for multi-objective optimization problems. In this method, the task criterion is formulated as a weighted sum of the objective functions where system designer assigns relative importance (weight) to each objective function. However, the task criterion obtained by this method lacks physical meaning. Misinterpretation of the theoretical and practical meaning of the weights can make the final solution unsatisfactory. Although there are many methods for choosing weights, a priori selection of weights does not necessarily guarantee the acceptability of the solution. Furthermore, it is impossible to obtain a solution on the non-convex regions of the Pareto frontier [7].

Thus, physical programming [8], is used to generate the task criterion. Using this method, the system designers only need to specify a preference structure for each objective instead of assigning a meaningless weight. This usually has more physical meaning and can better guarantee a satisfactory solution. Using physical programming, the system design can be put into a more flexible and natural framework. Another advantage of physical programming is that it can obtain the solutions in the non-convex regions of the Pareto frontier [7], [8].

3) Preferences Mapping: Using physical programming approach, objective functions are mapped into a preference space. The preference is a parameter which represents the extent of the designer's satisfaction. The task criterion is setup as the aggregation of the preferences of all the objective functions. Thus, instead of being physically meaningless, the criterion becomes a satisfaction factor of the solution.

The designer's expression of the preferences with respect to each objective function can be categorized into four different classes: smaller-is-better (1S), larger-is-better (2S), value-is-better (3S), range-is-better (4S) as in [7]. Each objective belongs to one of these classes. [7] provides a more in-depth analysis of physical programming.

The class functions for preference mapping should have several important properties such as nonnegativity, continuity and convexity. Further the preference value P at range intersections (e.g., Tolerable-Undesirable) is the same for all class types and objectives.

- they are nonnegative, continuous and convex
- the preference value P at the range intersection (for instance, Tolerable-Undesirable) is the same for all the class types and all the objectives.

4) Physical Programming Formulation: Based on the preference mapping of the objectives, physical programming problems can be stated as:

$$\min_{\mathbf{x}} Z = \sum_{i=1}^{M} P_i\left(g_i(\mathbf{x})\right) \quad (17)$$

subject to

$$\begin{array}{ll} g_i(\mathbf{x}) \leqslant v_{i5} & \text{(for class 1S)} \\ g_i(\mathbf{x}) \geqslant v_{i5} & \text{(for class 2S)} \\ v_{i5L} \leqslant g_i(\mathbf{x}) \leqslant v_{i5R} & \text{(for class 3S and 4S)} \end{array}$$

where **x** is the system variables, M is the number of the objective functions, $g_i(\mathbf{x})$ is the ith objective function, P_i is the preference class function of g_i.

C. PP Analyzer

The preference structure for each objective function must be specified to setup the global criterion function. However, due to changing task requirements, the initially specified preferences for the objectives may not represent the objective's desirability accurately at later time instants. Therefore, physical programming should be modified according to the task requirements.

A PP analyzer module is built to handle this problem. The PP analyzer takes the Task Indices as inputs, and generates a vector $\Xi = [\xi_1, \cdots, \xi_M]^T$ to represent the degree of designers' confidence about the preference structure specified for the objective function. It passes this confidence vector to the physical programming module which modifies its result according to the confidence parameters. The analyzer is formulated as:

$$\xi_i = (a_i + k_i \times index_i) \quad i = 1, \cdots, M \tag{18}$$

where a_i is a constant which represents the system designer's confidence about the original preference mapping function, k_i is a constant. The criterion generated by physical programming will be modified as:

$$Z = \sum_{i=1}^{M} \xi_i \times P_i\left(g_i(x)\right) \tag{19}$$

Equations (18) and (19) show that when $index_i$ increases, the

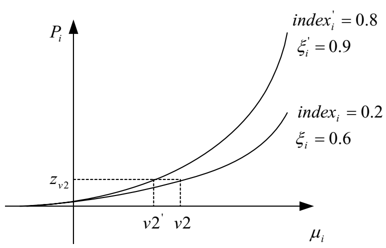


Fig. 3. Modification of the preference mapping by confidence factor.

preference value of the i^{th} objective function will increase. Thus, when the task has high preference on the requirement class i, the objective function D_i should have a higher preference for further minimization. Thus, the task requirement condition governs the path of optimization.

From another viewpoint, multiplying a factor with the preference mapping function can be treated as modifying the structure of the mapping. Taking the class 1S as an example (from Figure 3), if the preference value at the range intersection (say z_{v2} Desirable-Tolerable) is kept unchanged, the desirability ranges of the objective function change with the task index (confidence factor). In this figure, $index' > index$ will induce $v2' < v2$ which makes the Tolerant range closer to the ideal range.

Therefore, combining equations (13), (17) and (18), replace the g_i by D_i, and **x** by u, p, the optimal Hausdorff tracking problem is formulated as:

$$\min_{u,p} Z = \sum_{i=1}^{M} \xi_i \times P_i\left(D_i(u,p)\right)$$

subject to

$$\begin{array}{ll} -\alpha J \geqslant Cu & \\ D_i(u,p) \leqslant v_{i5} & \text{(for class 1S)} \\ D_i(u,p) \geqslant v_{i5} & \text{(for class 2S)} \\ v_{i5L} \leqslant D_i(u,p) \leqslant v_{i5R} & \text{(for class 3S and 4S)} \end{array} \tag{20}$$

where, α is a positive definite gain and J is the vector of shape functions described for a specific task. The system constraint equation $-\alpha J \geqslant Cu$ is derived from the Hausdorff tracking task controller derived in [2].

D. Optimal Solution Calculation

The next step is to find the optimal solution for the optimization problem specified by Equation (20). In this formulation the system constraint $-\alpha J \geqslant Cu$ takes the form of a linear inequality. We can use the generalized elimination method [9] to eliminate the system dependant variables: suppose that there exist matrices $A_{n\times m}$ and $B_{n\times(n-m)}$ such that $[A\ B]$ is non-singular. Usually, for the single target Hausdorff tracking task, $n = 6$ and $m = 1$. If matrices A and B satisfy $CA = I$ and $CB = 0$ then the system solution is formulated as:

$$u \leqslant -\alpha AJ + B\epsilon \tag{21}$$

where, ϵ denotes the Null space of the matrix C.

The reduced optimization problem is written as:

$$\min_{\epsilon} Z = \sum_{i=1}^{M} \xi_i \times P_i\left(D_i(u,p)\right) \tag{22}$$

subject to

$$\begin{array}{l} D_i(u,p) \leqslant v_{i5} \ \text{(for class 1S)} \\ D_i(u,p) \geqslant v_{i5} \ \text{(for class 2S)} \\ v_{i5L} \leqslant D_i(u,p) \leqslant v_{i5R} \ \text{(for class 3S and 4S)} \end{array} \tag{23}$$

$$\text{where,} \quad \begin{array}{ll} u & \leqslant -\alpha AJ + B\epsilon \\ p & = \bar{p} + u/f \end{array} \tag{24}$$

$\bar{p}$ is the initial value of the system variables p, f is the control frequency, ϵ is and arbitrary vector. Using an online pattern search algorithm [10], the system can easily obtain the optimal solution.

IV. Experimental Implementation and Results

The problem we are considering is the optimal task distribution for a pan-tilt camera mounted on a mobile robot tracking a single target depicted in Figure 1(b). The target is a human wearing a solid color shirt.

We will consider the redundancy in the robot and camera motion along the X direction. When the target moves along the X direction, either the robot can traverse linearly or the camera can pan. Hence there is a redundancy in the task execution. The model for the Hausdorff tracking task using the camera and the robot combination can be written as:

$$[C_1 C_2] \begin{bmatrix} v_x \\ w_y \end{bmatrix} \leqslant -\alpha J \tag{25}$$

where $u = [v_x\ w_y]^T$ is the velocity input vector to the robot and camera combination, α is a positive task level scalar gain and J is the shape function which indicates the error in accomplishing the task. The vector $C = [C_1\ C_2]$ describes the distribution of the task error between the two redundant axes. Equation (25) can be derived using the formula provided in [2].

The task indicators used are the energy consumed by the system and the current pan angle of the camera. The current pan angle of the camera affects how the translation in X by the mobile robot affects the task accomplishment. Hence, the objective functions considered for the multi-objective

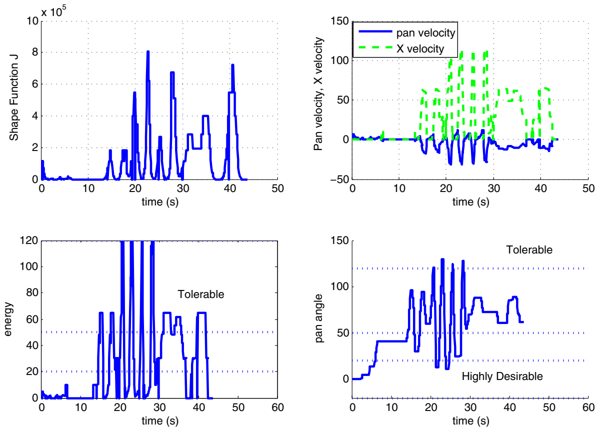
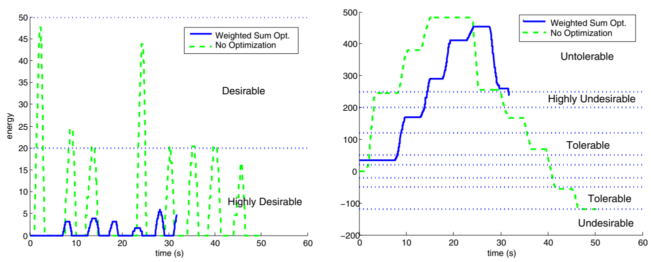


Fig. 4. Results of the tracking task with optimal task distribution using physical programming.



(a) The energy consumption.

(b) The pan angle.

Fig. 5. Task Criteria for no optimization, and weighted sum optimization.

optimization schemes also included energy minimization and pan angle minimization and were represented as:

$$D_1 = q_0^2 = (O - O_0)^2 \tag{26}$$

$$D_2 = \|v_x\| + \|w_y\| \tag{27}$$

where, O is the current pan angle value, O_0 is the desired pan angle value considered as zero pan, $\|v_x\|$ and $\|w_y\|$ are the horizontal X velocity and the angular Y velocity (pan velocity) respectively. Note that confidence parameters are built into the objective functions. For example when the pan angle is close to zero, we would like to reduce the weight for the objective function D_1 which is exactly what happens.

Figure 4 shows the results of the experiments carried out using physical programming optimization. The figure shows the plots of the shape function, the velocities in the X and pan directions and the objective function values for energy conservation and pan angle zeroing. Notice that the values of all the objective functions are always maintained at least in the tolerable range by sacrificing the performance of the other cost functions currently under consideration.

The same task was also carried out using a no-optimization scheme with a fixed task distribution between the two redundant axes and also using a weighted sum scheme. A weighted sum method with fixed weights without apparent meaning is also used in order to combine the various cost criteria. The cost criterion used is:

$$Z = q_0^2 + \frac{(\|v_x\| + \|w_x\|)}{2} \tag{28}$$

The two cost functions i.e., energy and pan-angle maintenance for the no-optimization method and the weighted sum method are shown in Figures 5(a) and 5(a) respectively.

For the no-optimization case which involves just a random selection of the Null space vector ϵ, it can be seen that in the absence of any optimization, the linear velocity transferred to the robot is very small and most of the velocity for task execution is allocated to the camera pan.

Using the weighted sum method, the various components comprising the task criterion lack physical meaning. Based on this observation we notice that even when the multi-objective optimization is enabled, the pan angle remains very close to the desired value while the energy conservation objective attains unacceptable values. It should be noted that various tolerance ranges, which are used for the physical programming approach, are not used in the weighted sum approach. Hence, despite the optimization being enabled on the various objectives, the values of the objective functions can attain unacceptable values.

V. CONCLUSION

This paper extends the Hausdorff tracking problem to incorporate an optimal control framework for executing multiple subtasks along with the target visibility task for a surveillance network. This optimization framework will allow the surveillance task designer to consider various sub-tasks in planning phase. Various sub-tasks can include minimizing the energy consumed by the network which will enhance the longevity of the deployed network sensors. The use of physical programming for finding a solution to the optimization problem is proposed in order to lend a tangible and physical interpretation for the subtasks being considered. Experimental results for a tracking task are presented which show the advantages of the physical programming approach over the regularly used weighted sum approach.

REFERENCES

[1] J. Cea, "Problems of shape optimal design," *Optimization of Distributed Parameter Structures vol. I and II*, pp. 1005–1087, 1981.

[2] A. Goradia, N. Xi, Z. Cen, and M. Mutka, "Modeling and design of mobile surveillance networks using a mutational analysis approach," in *Proceeding of the IEEE International Conference on Intelligent Robots and Systems*, 2005.

[3] J. Sokolowski and J.-P. Zolesio, *Introduction to Shape Optimization: Shape Sensitivity Analysis*, ser. Computational Mathematics. Springer-Verlag, 1991.

[4] L. Doyen, "Shape laypunov functions and stabilization of reachable tubes of control problems," *Journal of Mathematical Analysis and Applications*, vol. 184, pp. 222–228, 1994.

[5] J.-P. Aubin, "Mutational equations in metric spaces," *Set-Valued Analysis*, vol. 1, pp. 3–46, 1993.

[6] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.

[7] A. Mullur, C. Mattson, , and A. Messac, "New decision matrix based approach for concept selection using linear physical programming."

[8] A. Messac, "From dubious construction of objective functions to the application of physical programming," *AIAA Journal*, vol. 38, no. 1, pp. 155–163, Jan 2000.

[9] R. Fletcher, *Practical Methods of Optimization, Second Edition*. Anchor Brendon Ltd., Tiptree, Essex.

[10] B. S. Gottfried and J. Weisman, *Introduction to Optimization Theory.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1973.