Exact and Distributed Algorithms for Collaborative Camera Control *

Dezhen Song[1], A. Frank van der Stappen[2], and Ken Goldberg[3]

[1] IEOR Department, UC Berkeley, USA
[2] ICS Department, Utrecht University, Netherlands
[3] IEOR and EECS Department, UC Berkeley, USA

Abstract. We propose the ShareCam Problem: controlling a single robotic pan, tilt, zoom camera based on simultaneous frame requests from n online users. To solve it, we propose a new piecewise linear metric, Intersection Over Maximum (IOM), for the degree of satisfaction for each users. To maximize overall satisfaction, we present several algorithms. For a discrete set of m distinct zoom levels, we give an exact algorithm that runs in $O(n^2m)$ time. The algorithm can be distributed to run in $O(nm)$ time at each client and in $O(n \log n + mn)$ time at the server.

1 Introduction

Consider a robotic camera at a compelling location such as the Sydney boat harbor, United Nations, Academy Awards, or inside the International Space Station. There are $n >> 1$ viewers, and one camera. The camera frame is determined by pan, tilt, and zoom parameters that can be changed to observe details of the scene. The existing control method is based on queueing, where users have to wait patiently for their turn to operate the camera [20]. Can we eliminate the queue and allow many users to share control of the camera simultaneously?

We are developing network-based applications for education, journalism and entertainment where many users share control of a single physical resource. "Sharecam", is an example of Collaborative Telerobotics, where the camera is a telerobot with 3 degrees of freedom. In the taxonomy proposed by Chong et al. [9], this is a Multiple Operator Single Robot (MOSR) system. Our research is motivated by applications where groups of users desire simultaneous access to a single robotic resource. Inputs from each user are combined to generate a single control stream for the robot. There can be benefits to such collaboration: teamwork is a key element in education at all levels [10,36] and an ensemble of users may be more reliable than a single (possibly malicious) user.

As illustrated in Figure 1, the user views two windows, one for user input with a fixed image and the other with a variable image based on collective

* This work was supported in part by the National Science Foundation under IIS-0113147 and by Intel Corporation. For more information please contact dzsong@ieor.berkeley.edu or goldberg@ieor.berkeley.edu.

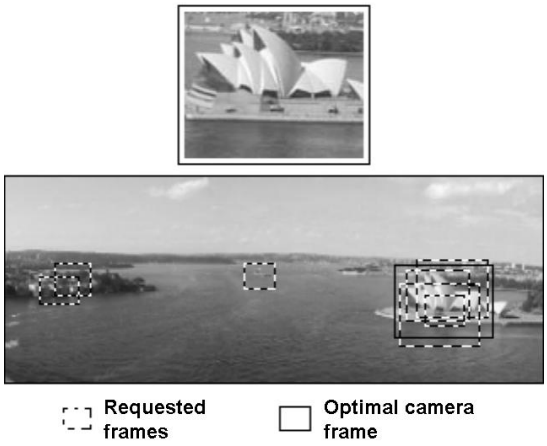

Fig. 1. Sharecam interface. Each Internet-based user views two image windows. The lower window is a fixed image of the camera's reachable range of view. Each user requests a camera frame by positioning a dashed rectangle in the lower window. Based on these requests, the algorithm computes an optimal camera frame (shown with solid rectangle), moves the camera accordingly, and displays the resulting live image in the upper window.

output. The input is a set of requested camera frames specified as desired fixed aspect-ratio iso-oriented rectangles from n users. The output is a single camera frame based on all inputs. We propose a new metric for user "satisfaction" based on how a user's requested frame compares with a candidate camera frame.

The metric for user satisfaction is proportional to the common area of the candidate frame and the requested frame. In addition, it is inversely proportional to the ratio of the sizes of the candidate and the request, to discourage excessively large candidates, such as an enclosing rectangle of all requests.

Finding the camera frame that maximizes total satisfaction is a non-linear optimization problem. We define a notion of "virtual corners" and prove that a global maximum must coincide with one of the virtual corners. We then present algorithms and complexity analysis. For n users and m zoom levels, the virtual corner search algorithm runs in time $O(n^2m)$. We then present distributed versions of the virtual corner search algorithm which run in time $O(n \log n + mn)$ on the server and $O(mn)$ on each client without loss of accuracy.

2 Related Work

The Internet provides a low-cost and widely-available interface that can make physical resources accessible to a broad range of participants. Online robots,

controllable over the Internet, are an active research area. In addition to the challenges associated with time delay, supervisory control, and stability, online robots must be designed to be operated by non-specialists through intuitive user interfaces and to be accessible 24 hours a day; see [21,37,23,14] for examples of recent projects.

Chong et al. [9] proposed the following taxonomy for teleoperation systems: Single Operator Single Robot (SOSR), Single Operator Multiple Robot (SOMR), Multiple Operator Multiple Robot (MOMR). and Multiple Operator Single Robot (MOSR). Most online robots are SOSR, where control is limited to one operator at a time. One precedent of an online MOSR system is described by McDonald, Cannon, and colleagues [7,29]. In their work, several users assist in waste cleanup using Point-and-Direct (PAD) commands. Users point to cleanup locations in a shared image and a robot excavates each location in turn.

In [13,12], Goldberg and Chen described an Internet-based MOSR system that averaged multiple human inputs to simultaneously control a single industrial robot arm. In [15] Goldberg, Song, et al. propose the "Spatial Dynamic Voting" (SDV) interface. The SDV collects, displays, and analyzes a sequence of spatial votes from multiple online operators at their Internet browsers. The votes drive the motion of a single mobile robot or human "Tele-Actor".

We formulate the collaborative control problem as a nonlinear optimization problem with a non-differentiable objective function. The structure of the problem is closely related to the planar $p-$center problem, which has been proved to be NP-complete by Megiddo and Supowit [30]. Using a geometric approach, Eppstein [11] found an algorithm for the the planar 2-Center problem in $O(n \log^2 n)$. Halperin et al. [18] gave an algorithm for the 2-center problem with m obstacles that runs in randomized expected time $O(m \log^2(mn) + mn \log^2 n \log(mn))$.

In almost all nonlinear mathematical programming approaches, a constrained optimization problem is converted to a series of unconstrained problems using barrier or penalty methods. Line search is then used to solve the unconstrained optimization problems. Although there are many different ways of guiding search direction and step size, most of these methods are based on derivatives [34].

Rectangle-related problems in computational geometry include range searching and rectangle intersection. Agarwal and Erickson [1] provide a review of geometric range searching and its related topics. Grossi and Italiano [16,17] proposed the cross-tree data structure, a generalized version of balanced tree, to speed up range searching in high-dimensional space.

Kapelios et al. [25] developed an algorithm that reports the set of iso-oriented rectangles that intersect a query rectangle but do not enclose it and do not have one of their vertices inside it. They give solutions for unrestricted and restricted universe (grid) for R^d. Mount et al. [32] study, for a given pair

of polygons P and Q, how to get the area of overlap of P and $Q + t$ as a function of the translation vector t.

It is also possible to view camera frame selection as a clustering problem [22,24]. The user satisfaction metric function we defined later in the paper is based on the resemblance and containment relationship between users' requested camera frames and real camera frame. The Symmetric Difference (SD) and "Intersection Over Union"(IOU) are well-known similarity metrics [3,5,39]. The comparison of these metrics and our metric will be discussed later in the paper.

There is also a connection with distributed manipulation. One branch of distributed manipulation uses potential fields defined as "potential-per-unit-area" acting on an object [4,31]. It is possible to interpret the satisfaction function as a special "lifted" potential field with some modifications.

Distributed computation has been used for sensor processing [33], multi-actuator control, and multi-robot systems. Sagawa et al. [38] developed a parallel algorithm to merge a set of range images into a volumetric implicit surface image representation, which is converted to a surface mesh. Safaric et al. [27] designed a distributed control system for an active surface device. The active surface device is a massive parallel micro-actuator array that can generate a pressure field on a planar surface. Applications of distributed algorithms include motion planing [6,35], localization [19,33], and task allocation [2,8].

In independent work, Kimber, Liu, Foote et al describe a multi-user robot camera in [26,28]. As we do in Sharecam, they formulate the frame selection for multiple simultaneous requests as an optimization problem based on position and area of overlap. To solve it, they propose an approximation based on comparing the bounding box of all combinations of user frames. This algorithm requires exponential time and does not provide formal bounds on approximation error.

3 Problem definition

In this section we formulate the optimization problem: finding the camera frame that maximizes total user satisfaction.

3.1 Input and assumptions

Let ϕ be a vector of system parameters. In the Sharecam system, $\phi = [x, y, z]^T$, where x, y specify the center point of the input rectangle corresponding to pan and tilt, and z specifies size of the rectangle, which can be used to control zoom. Since the camera has a fixed aspect ratio of 4:3, the length of the rectangle is $4z$ and width of the rectangle is $3z$. At each time increment, user i requests a desired frame, ϕ_i. Given requests from n users,

the system computes a single global frame ϕ^* that will best satisfy the set of requests.

Let Θ be the set of all admissible $[x, y]$ pairs. Let Z be a small set of attainable discrete zoom levels. Hence the solution space is:

$$\Phi = \Theta \times Z = \{[x, y, z] | [x, y] \in \Theta, z \in Z\}.$$

3.2 Definition of a metric for user satisfaction

Recall that ϕ_i is the frame requested by user i, and let $\phi = [x, y, z]^T$ be a candidate camera frame. We define a scalar $s_i \in [0, 1]$ as the level of "satisfaction" that user i receives. User i should get no satisfaction if the candidate frame does not intersect ϕ_i, so $s_i = 0$ when $\phi \cap \phi_i = \emptyset$. User i should be perfectly satisfied when the candidate frame is identical to ϕ_i, so $s_i = 1$ when $\phi = \phi_i$. We define the satisfaction of user i provided by the candidate frame ϕ to be Intersection Over Maximum (IOM),

$$s_i(\phi_i, \phi) = p_i / \max(a, a_i)$$

where a is the area of the candidate ϕ ($a = 12z^2$), frame a_i is the area of frame ϕ_i ($a_i = 12z_i^2$), and p_i is the area of overlap between ϕ_i and ϕ. Furthermore,

$$s_i(\phi_i, \phi) = p_i / \max(a, a_i) = (p_i/a_i) \min((z_i/z)^2, 1)$$

is a special case of Generalized Intersection Over Maximum metric (GIOM) for $b = 2$,

$$s_i(\phi_i, \phi) = (p_i/a_i) \min((z_i/z)^b, 1)$$

where b is a small positive number.

If z is bigger, the candidate frame will be bigger. A sufficiently large z can define a candidate frame that covers all requested frames: $p_i/a_i = 1$ for $i = 1, ..., n$. However, user satisfaction is not necessarily high because a user wants to see the requested frame at a desired zoom level. The term $\min((z_i/z)^b, 1)$ characterizes this wish: it reaches its maximum of 1 if the candidate frame is the same or smaller than the requested frame. The $s_i \in [0, 1]$ is normalized.

During experiments, we found that if b is large, we have to pay more for increasing the size of the candidate frame in the objective function value. Therefore, the optimal frame tends to be small. We want to avoid cases where the optimal frame is too small such that it can only cover the intersected area among the requested frames. We thus set $b = 1$ in our settings to encourage the optimal frame to be a trade-off between the small intersected area of the requested frames and the big union area of the requested frames. Figure 2 shows five examples of requested frames and the corresponding satisfaction, for $b = 1$.

The total satisfaction for n users is

$$s(\phi) = \sum_{i=1}^{n} s_i(\phi_i, \phi). \tag{1}$$

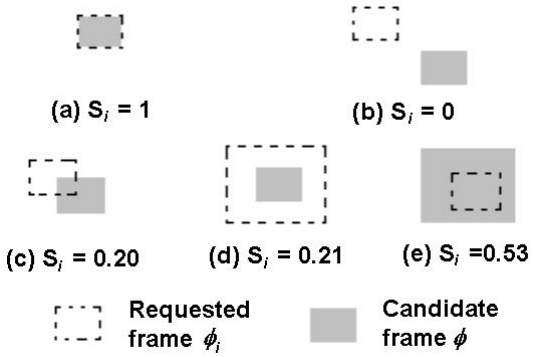

Fig. 2. Examples of the satisfaction metric for user i and candidate frame ϕ.

We want to find $\phi^* = \arg\max_\phi s(\phi)$, the frame that maximizes total satisfaction. To describe the objective function with respect to x, y, and z,

$$s(x, y, z) = s(\phi).$$

Equation 1 can be extended to a weighted sum format at time t:

$$s(\phi, t) = \sum_{i=1}^{n} \alpha_i(t) s_i(\phi_i(t), \phi(t)) \tag{2}$$

where the weight $\alpha_i(t)$ for user i is a function of the user's previous unsatisfaction levels: $u_i(t) = 1 - s_i(\phi_i(t), \phi(t))$. An example of $\alpha_i(t)$ is,

$$\alpha_i(t) = \sum_{k=0}^{t-1} \frac{u_i(k)}{2^{t-1-k}}$$

and its recursive formulation is very simple,

$$\alpha_i(t) = u_i(t-1) + \alpha_i(t-1)/2$$

If user i is not satisfied in previous runs, his/her $\alpha_i(t)$ will increase in the objective function. This formulation can avoid the situation that some user may not be satisfied all the time. From algorithm point of view, the difference between equation 1 and equation 2 is little. To simplify the discussion, we will use the objective function described by equation 1 in the rest of the paper.

3.3 Properties of the satisfaction metric

We show that the objective function s is nonsmooth and piecewise linear in both x and y. In addition we compare it to the intersection-over-union metric.

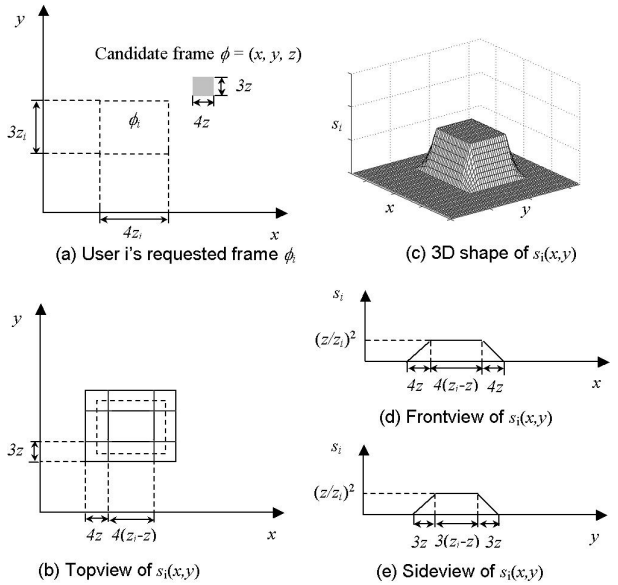

Fig. 3. Satisfaction function, $s_i(x, y)$, for a given candidate frame and $b = 1$. Since $z \leq z_i$ is given, we can move the candidate frame (gray rectangle) around the user i's requested frame to observe how $s_i(x, y)$ changes. The function is a plateau-like shape with a maximum height of $p_i/a_i = 12z^2/12z_i^2 = (z/z_i)^2$. The function consists of 5 planes and 4 quadratic surfaces at the corners.

Nonsmoothness. Recall that the objective function is

$$s(\phi) = \sum_{i=1}^{n} (p_i/a_i) \min((z_i/z)^b, 1).$$

Since we solve this problem for each attainable zoom level z, we treat z as a constant. The objective function becomes a function of the center point of the candidate frame,

$$s(x, y) = \sum_{i=1}^{n} \omega_i p_i(x, y) \tag{3}$$

where $\omega_i = (1/a_i) \min((z_i/z)^b, 1)$ is a constant for each user. We know that $p_i(x, y)$ is the area of the intersection of the requested frame of user i and the candidate frame (x, y, z). Therefore, the maximum value of $p_i(x, y)$ is $\min\{a_i, a\}$. This property determines that the shape of user i's satisfaction function is plateau-like. Figure 3 shows the shape of $s_i(x, y)$ given $z \leq z_i$, i.e. the candidate frame is smaller than the requested frame of user i. Note that s_i is non-differentiable with respect to x and y so we cannot use derivative-based approaches to solve this problem.

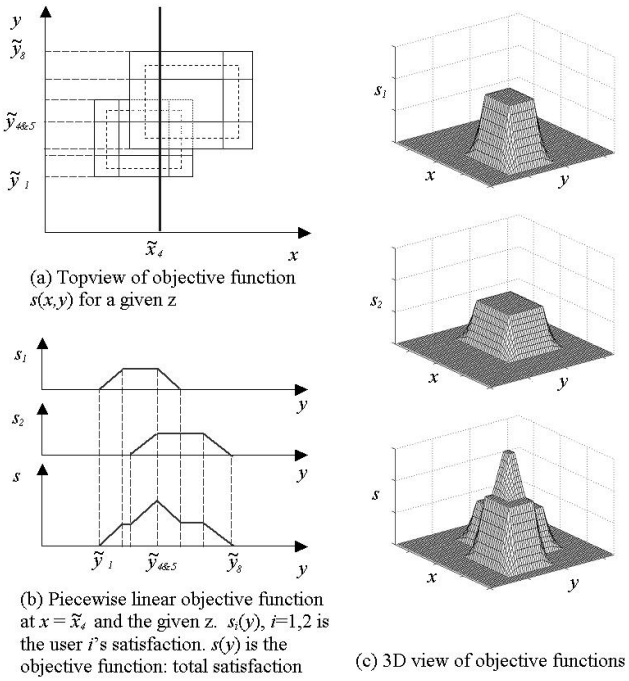

Fig. 4. Satisfaction function $s_i(y)$ for two-users. Ordered sets $\{\tilde{y}_k\}$ and $\{\tilde{x}_k\}$, $k = 1, ..., 8$ are corresponding to horizontal and vertical edges of plateaus. Note that $\tilde{y}_4$ and $\tilde{y}_5$ overlap in this case.

Piecewise linearity in x and y. Since all requested frames and the candidate frame are iso-oriented rectangles, the shape of any intersection between them is also a rectangle with its edges parallel to either x axis or y axis. Thus the term $p_i(x, y)$ in equation 3 is either 0 or the area of the rectangle formed by intersection between ϕ_i and $\phi = (x, y, z)$. This yields a nice property: the $p_i(x, y)$ is piecewise linear with respect to x if we fix y, and piecewise linear with respect to y if we fix x. Since the total satisfaction metric $s(x, y)$ is a linear combination of $p_i(x, y)$, $i = 1, ..., n$, it has the same property. Figure 4 shows an example for a case with two requested frames.

Comparison to other metrics. In pattern recognition and computational geometry standard similarity metrics are Symmetric Difference (SD) and Intersection Over Union (IOU)[3,5,39]. For a requested frame ϕ_i and a candidate frame ϕ, the SD metric is

$$SD = \frac{Area(\phi_i \cup \phi) - Area(\phi_i \cap \phi)}{Area(\phi_i \cup \phi)}.$$

The intersection-over-union metric is

$$IOU = \frac{Area(\phi_i \cap \phi)}{Area(\phi_i \cup \phi)} = 1 - SD.$$

Compared with IOU, our satisfaction metric has similar properties: (i) both attain their minimum value of 0 if and only if $\phi \cap \phi_i = \emptyset$, (ii) both attain their maximum value of 1 if and only if $\phi = \phi_i$, (iii) both are proportional to the area of $\phi \cap \phi_i$, and (iv) both depend—albeit differently—on the sizes of ϕ and ϕ_i. However, the intersection-over-union metric is not piecewise linear in x or y.

4 Algorithms

In this section we present algorithms for two versions of the problem of finding the frame that maximizes total satisfaction. We start with a version in which the pan (x) as well as the tilt (y) are restricted to a discrete set of equally-spaced values. Subsection 4.1 describes a brute-force algorithm for this discrete version of the problem. In Subsection 4.2 we allow the pan and tilt to vary continuously. This more general continuous version allows for an efficient exact algorithm. The algorithm exploits a geometric characteristic of the optimal solution (captured in the notion of a "virtual corner"). The exact algorithm can also be distributed across the client machines and the server. The distributed algorithm is given in Subsection 4.3.

4.1 Algorithms for discrete pan and tilt

Theorem 1 *Let w be the width (in pixels) of the camera's total pan range, and h be the height (in pixels) of the camera's total tilt range. We can solve the discrete pan and tilt problem in $O(whmn)$.*

Proof. A brute force search for finding $\phi^* = \arg\max_\phi s(\phi)$ evaluates all whm candidate points. According to Equation 1, it takes $O(n)$ computing time to determine the satisfaction for a single candidate frame ϕ. The total amount of computation of the algorithm is $O(whmn)$.

Although this is linear in n, the constants are large (typically $w = 600, h = 200$).

4.2 An algorithm for continuous pan and tilt

We now focus on the more general problem where the camera's pan and tilt may vary continuously.

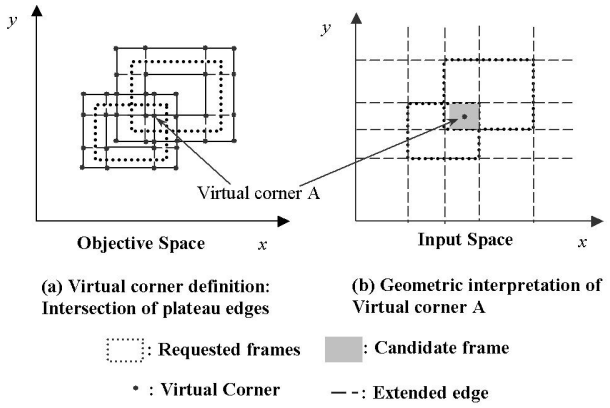

Fig. 5. Virtual corners and a geometric interpretation for two requested frames. A virtual corner corresponds to a candidate frame that has one corner at the intersection of one extended vertical edge of a requested frame and one horizontal extended edge of a requested frame.

Virtual corners. As shown in Figure 3, the $s_i(x, y)$ is a plateau-like function. For n users, there are n plateaus. Each plateau consists of 9 facets: 1 top plane, 4 side planes, and 4 quadratic surface at corners. There are two vertical boundaries and two horizontal boundaries at the bottom (bounding the entire plateau), the same numbers of similar edges at the top (bounding the plateau's flat top), and eight boundaries separating side planes and corner quadratic surfaces (see Figure 5a).

A *virtual corner* is an intersection between any two boundaries, which includes both intersections of facet boundaries induced by a single plateau or by two distinct plateaus. Since all plateaus are iso-oriented, one of the extensions has to be horizonal and the other has to be vertical. For n requested frames, there are $O(n^2)$ virtual corners. Figure 5b shows the geometric interpretation of virtual corner in the input space. If we map the virtual corner in the objective space back to input space, we see that the virtual corner corresponds to a candidate frame that has one corner overlapping with the intersection of two extensions of edges of requested frames. What makes a virtual corner important is the following theorem.

Lemma 1 *At least one optimal frame is centered at a virtual corner.*

Proof. Let $\phi^* = [x^*, y^*, z^*]$ be an optimal solution. As discussed earlier, for a fixed z and x, the objective function $s(y)$ is piecewise linear. So the optimum must be at a vertex $y = \tilde{y}$ such that $s(x^*, \tilde{y}, z^*) = s(x^*, y^*, z^*)$. We also know that line $y = \tilde{y}$ in (x, y) plane is one of the horizontal facet boundaries of the plateaus. Similarly, we can find another optimal frame $[\tilde{x}, \tilde{y}, z^*]$, where line $x = \tilde{x}$ is one of the vertical facet boundaries of the plateaus. Therefore, the optimal frame $[\tilde{x}, \tilde{y}, z^*]$ is centered at a virtual corner $(\tilde{x}, \tilde{y})$.

Brute force approach. Based on the theorem 1, we can solve the optimization problem by simply checking all combinations of zoom levels and corresponding virtual corners. We need to evaluate the objective function for each of the $O(n^2)$ virtual corners and repeat this for each of the m zoom levels. It takes $O(n)$ time to evaluate a candidate frame ϕ. Therefore, the brute force algorithm runs in $O(n^3m)$. However, we can do better if (for a fixed zoom level) we handle all virtual corners with the same x-coordinate consecutively in order of increasing y-coordinate, and take advantage of the fact that the objective function only changes slightly between two consecutive virtual corners.

Efficient traversal of virtual corners. For n requested frames, we have $4n$ horizontal plateau facet boundaries $\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_{4n}\}$ and $4n$ vertical plateau facet boundaries $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_{4n}\}$ for plateaus. We can reduce the computation complexity from $O(n^3m)$ to $O(n^2m)$. Recall that $\phi_i = [x_i, y_i, z_i]$, $i = 1, ..., n$ are the requested frames.

$s^* = 0$, $O(1)$
Sort $\{y_i + 1.5z_i\}$, $i = 1, ...n$ $O(n \log n)$
Sort $\{y_i - 1.5z_i\}$, $i = 1, ...n$ $O(n \log n)$
For each zoom level z (m in total)
i) Compute vertical extended plateau edges $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_{4n}\}$ $O(n)$
For each user i, $i = 1, ..., n$,
$\tilde{x}_{4i-3} = x_i - 2(z_i + z)$,
$\tilde{x}_{4i-2} = x_i - 2(z_i - z)$,
$\tilde{x}_{4i-1} = x_i + 2(z_i - z)$,
$\tilde{x}_{4i} = x_i + 2(z_i + z)$,
ii) Compute the sorted sequence $\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_{4n}\}$, $O(n)$
For each i, $i = 1, ..., n$
$\tilde{y}_{4i-3} = (y_i - 1.5z_i) + 1.5z$,
$\tilde{y}_{4i-2} = (y_i - 1.5z_i) - 1.5z$,
$\tilde{y}_{4i-1} = (y_i + 1.5z_i) - 1.5z$,
$\tilde{y}_{4i} = (y_i + 1.5z_i) + 1.5z$,
Merge the 4 ordered sequences:
$\{\tilde{y}_{4i-3}\}$, $\{\tilde{y}_{4i-2}\}$, $\{\tilde{y}_{4i-1}\}$, and $\{\tilde{y}_{4i}\}$, $i = 1, ..., n$
to get the ordered sequence $\{\tilde{y}_1, \tilde{y}_2 ..., \tilde{y}_{4n}\}$,
where $\tilde{y}_1$ is the smallest.
iii) For $x = \tilde{x}_i$, $i = 1, ..., 4n$,
$s = \max_y \ s(\tilde{x}_i, y, z)$,
if $s > s^*$ then $s^* = s, x^* = \tilde{x}_i, y^* = y, z^* = z$.
Output s^* as optimal objective function value and (x^*, y^*, z^*) as optimal frame.

In step iii, we traverse the vertical facet boundaries of the plateaus one by one. For each vertical edge, we find the maximum by forcing the candidate

frame to center at it. Using Theorem 1, we know that this procedure will find an optimal solution. It remains to show how much time is required to solve the resulting problem of finding

$$\max_y \ s(x, y, z)$$

for given x and z. This special optimization problem can be solved in $O(n)$ with a sorted sequence $\{\tilde{y}_1, \tilde{y}_2..., \tilde{y}_{4n}\}$. The objective function is a "summation" of n plateaus, which is shown in Figure 4. For fixed x and z, this piecewise linear function only changes slope at $\{\tilde{y}_i\}$, $i = 1, ..., 4n$. For each vertex $\tilde{y}_i$, we know how much the slope will change after crossing the vertex. We can find the maximum objective value by walking over all ordered vertices $\{\tilde{y}_i\}$ from the one side to the other side on the line $x = \tilde{x}_i$. This process only takes $O(n)$. Therefore, step iii) of the algorithm will take $O(n^2)$ and the following theorem is true.

Theorem 2 *We can solve the Sharecam problem in time $O(n^2m)$ for n users and m zoom levels.*

4.3 A distributed algorithm

In the system, n is the number of users online, which is also the number of computers connecting to our server. The larger the value of n, the more computation power we have in our system. This suggests that a distributed computing strategy can further improve computational speed. The algorithm described in the previous section can be divided into client and server components.

Server algorithm. The server should do the following.

i). Send all requested frames ϕ_i, $i = 1, ..., n$ to all clients,
ii). Sort sequence $\{y_i + 1.5z_i\}$ and sequence $\{y_i - 1.5z_i\}$, $i = 1, ...n$ and send them to all clients, $O(n \log n)$
iii). Wait until all clients send their solutions $\{s_1^*, ..., s_n^*\}$ back.
Pick the largest. $O(n)$

Client Algorithm Let us assume that $\phi_i = (x_i, y_i, z_i)$ is client i's requested frame. After client i receives the data from the server, it executes the following algorithm.

$s_i^* = 0$
For each zoom level z
i) Compute the sorted sequence $\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_{4n}\}$, $O(n)$
(Same as the centralized version.)

ii)a. $s_1 = \max_y \ s(x_i - 2(z_i + z), y, z)$, $O(n)$
if $s_1 > s_i^*$ then $s_i^* = s_1, x_i^* = x_i - 2(z_i + z), y_i^* = y, z_i^* = z$.
ii)b. $s_2 = \max_y \ s(x_i - 2(z_i - z), y, z)$, $O(n)$
if $s_2 > s_i^*$ then $s_i^* = s_2, x_i^* = x_i - 2(z_i - z), y_i^* = y, z_i^* = z$.
ii)c. $s_3 = \max_y \ s(x_i + 2(z_i - z), y, z)$, $O(n)$
if $s_3 > s_i^*$ then $s_i^* = s_3, x_i^* = x_i + 2(z_i - z), y_i^* = y, z_i^* = z$.
ii)d. $s_4 = \max_y \ s(x_i + 2(z_i + z), y, z)$, $O(n)$
if $s_4 > s_i^*$ then $s_i^* = s_4, x_i^* = x_i + 2(z_i + z), y_i^* = y, z_i^* = z$.
iii) Send s_i^* and (x_i^*, y_i^*, z_i^*) to server.

As we can see from the algorithm, the server runs at $O(n \log n + mn)$ and each client runs at $O(nm)$. The following theorem holds,

Theorem 3 *We can distribute the Sharecam algorithm among the server and clients resulting in a running time of* $O(n \log n + mn)$.

One can also see from the algorithm that the speed of computation is limited by the slowest client. One idea is to set a timeout for clients and have the server run the computation for clients that do not respond in time.

5 Results

We have implemented the algorithms on a PC with 950Mhz AMD Athlon CPU and 1GB memory. The machine runs under Redhat Linux 7.1. The algorithm is programmed in both matlab and Java.

Figure 6 shows the results for four different scenarios. As we can see from Figure 6(a) and (b), the optimal frame does not necessarily have one corner overlapping with a corner of a requested frame. However, one corner of the optimal frame does overlap with one of the virtual corners. Figure 6(b) has three requested frames exactly the same as those in (a) and one more big requested frame. It is interesting to see how the optimal frame changes after the big requested frame joined in the system. Figure 6(c) shows that if all input rectangles fall far way from each other, the algorithm functions as a chooser, which selects one input rectangle as the output. Since the algorithm searches optimum bottom-up, it picks the lowest requested frame as the solution. Figure 6(d) shows that a large requested frame does not necessarily yield a large optimal frame.

Figure 7 shows the relationship between the optimal frame size and the choice of the b value in GIOM satisfaction metric. It confirms that our analysis in Subsection 3.2: the bigger the b is, the smaller the optimal camera frame is. If $b \to 0^+$, then the optimal camera frame becomes the smallest frame that contains all request frame: $p_i = a_i \ \forall i$. If $b \to \infty$, then the optimal frame will converge to the rectangle area that most of requested frame insect each other. The parameter b allows us to find the best tradeoff between union and intersection.

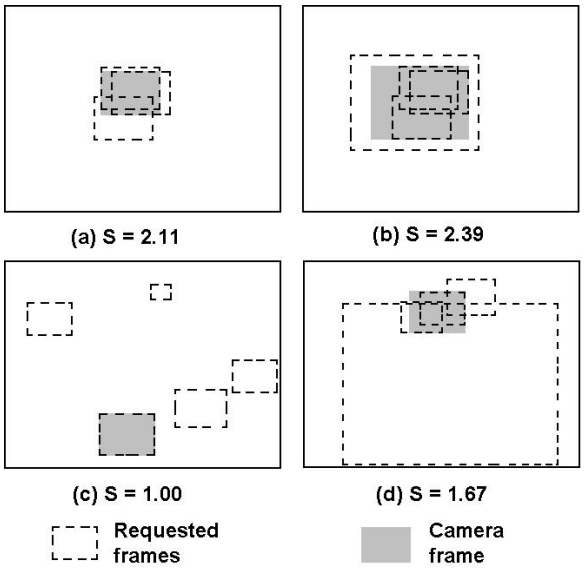

Fig. 6. Sample outputs of algorithm.

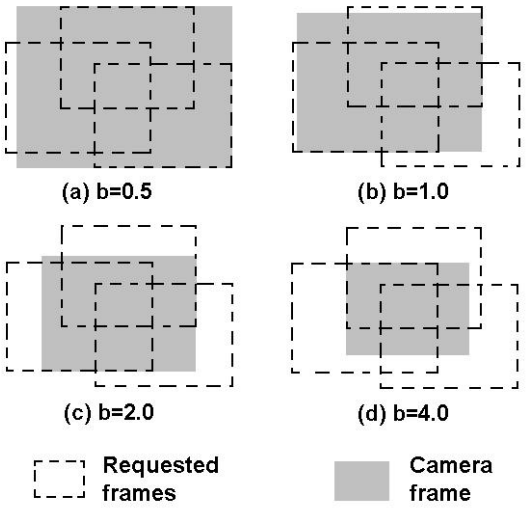

Fig. 7. Relationship between the optimal frame size and the choice of the b value in GIOM satisfaction metric in Subsection 3.2.

6 Conclusions and Future Work

This paper considers a new problem allowing multiple users to share control of a single remote robotic camera. Each user requests a desired camera frame by drawing a rectangle over a static global image. The problem is to find a camera frame that maximizes the overall user satisfaction. We define a new metric for user satisfaction and study algorithms for solving the nonlinear optimization problem.

We reviewed related work and studied properties of the objective function and inherent constraints on the location of extremal points. We defined "virtual corners" and proved that a global maximum must coincide with one of the virtual corners. We then presented algorithms and complexity analysis. For n users and m zoom levels, we described an efficient algorithm based on grouping and sorting of virtual corners running in time $O(n^2m)$. Finally we presented a distributed version of this algorithm that runs in time $O(n \log n + mn)$ on the server and $O(mn)$ on each client.

Unlike computing with multiple processors in a single supercomputer, distributed computing over the Internet requires input from a variety of heterogenous processors, each with different and varying communication delays. We are interested in distributed algorithms that optimize performance under such uncertainties.

In future work, we will also consider versions of the problem with continuous zoom levels ($m = \infty$). Preliminary results suggest that we can solve this in $O(n^3 \log n)$ time. We are also working on an non-uniform grid-based approximation algorithm. It employs a Branch and Bound-like approach and can solve the problems with continuous pan, tilt, and zoom in linear time. We will also consider extensions to p frames, with i) p sequential views from one camera or ii) from p different cameras. Scenario i) is also a scheduling problem. Scenario ii) is related to the p-center problem.

A preliminary version of the Sharecam system went online in Sep. 2002 at http://tele-actor.net/sharecam.

Acknowledgments

Thanks to A. Pashkevich, V. Koltun, D. Halperin, A. Lim, S. Rao, J. Luntz, and S. Har-Peled, for insightful discussions and feedback. Thanks to M. Faldu, F. Hsu, and W. Guan for their contributions to the ShareCam system, and K. "Gopal" Gopalakrishnan, Ron Alterovitz, A. Levandowski, M. Mckelvin, A. Ho, and J. Vidales for helpful suggestions.

References

1. P. Agarwal and J. Erickson. Geometric range searching and its relatives. In B. Chazelle, J. E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry, volume 23 of Contemporary Mathematics.*

2. W. Agassounon, A. Martinoli, and R. Goodman. A scaleable distributed algorithm for allocating workers in embedded systems in embedded systems. In *IEEE Systems, Man and Cybernetics Conference*, 2001.
3. A.Z.Broder. On the resemblance and containment of documents. In *SEQS: Sequences '91*, 1998.
4. K. F. Bohringer and H. Choset, editors. *Distributed Manipulation*. Kluwer Academic Publishers, 2000.
5. A. Z. Broder, M. Charikar, A.M. Frieze, and M.Mitzenmacher. Min-wise independent permutations. *Journal of Computer and System Sciences*, 60(3):630–659, 2000.
6. Z. Bulter, S. Byrnes, and D. Rus. Distributed motion planning for modular robots with unit-compressible modules. In *IEEE/RSJ International Conference on Intelligent Robots and System (IROS)*, 2001.
7. D. J. Cannon. *Point-And-Direct Telerobotics: Object Level Strategic Supervisory Control in Unstructured Interactive Human-Machine System Environments*. PhD thesis, Stanford Mechanical Engineering, June 1992.
8. R. Chen, Z. Wu, Y. Wang, and D. Tan C. Dong. Cooperative assembly algorithm of multiple distributed agent based robotic system. In *30th International Symposium on Robotics*, 1999.
9. N. Chong, T. Kotoku, K. Ohba, K. Komoriya, N. Matsuhira, and K. Tanie. Remote coordinated controls in multiple telerobot cooperation. In *IEEE International Conference on Robotics and Automation*, April 2000.
10. C. H. Crouch and E. Mazur. Peer instruction: Ten years of experience and results. *Americian Journal of Physices*, 69(9), 2001.
11. D. Eppstein. Fast construciton of planar two-centers. In *Proc. 8th ACM-SIAM Sympos. Discrete Algorithms*, 1997.
12. K. Goldberg and B. Chen. Collaborative control of robot motion: Robustness to error. In *International Conference on Intelligent Robots and Systems (IROS)*, 2001.
13. K. Goldberg, B. Chen, R. Solomon, S. Bui, B. Farzin, J. Heitler, D. Poon, and G. Smith. Collaborative teleoperation via the internet. In *IEEE International Conference on Robotics and Automation (ICRA)*, April 2000.
14. K. Goldberg and R. Siegwart, editors. *Beyond Webcams: An Introduction to Online Robots*. MIT Press, 2002.
15. K. Goldberg, D. Song, Y. Khor, D. Pescovitz, A. Levandowski, J. Himmelstein, J. Shih, A. Ho, E. Paulos, and J. Donath. Collaborative online teleoperation with spatial dynamic voting and a human "tele-actor". In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2002.
16. R. Grossi and G. F. Italiano. Efficient cross-trees for external memory. In James Abello and Jeffrey Scott Vitter, editors, *External Memory Algorithms and Visualization*, pages 87–106. American Mathematical Society Press, Providence, RI, 1999.
17. R. Grossi and G. F. Italiano. Revised version of "efficient cross-trees for external memory". Technical Report TR-00-16, 10 2000.
18. D. Halperin, M. Shairir, and K. Goldberg. The 2-center problem with obstacles. *Journal of Algorithms*, 32, 2002.
19. A.T. Hayes, A. Martinoli, and R.M. Goodman. Swarm robotic odor localization. In *IEEE/RSJ International Conference on Intelligent Robots and System (IROS)*, 2001.

20. **http://www.x-zone.canon.co.jp/WebView-E/all/list.htm**.
21. H. Hu, L. Yu, P. Tsui, and Q. Zhou. Internet-based robotic systems for teleoperation. *Assembly Automation Journal*, 21(2), 2001.
22. A.K Jain, M.N. Murty, and P.J. Flyn. Data clustering: A review. *ACM Computing Surveys*, 31(3), 1999.
23. S. Jia and K. Takase. A corba-based internet robotic system. *Advanced Robotics*, 15(6), 2001.
24. D.E. Johnson. *Applied Multivariate Methods for Data Analysis*. Duxbury Press, 1998.
25. V. Kapelios, G. Panagopoulou, G. Papamichail, S. Sirmakessis, and A. Tsakalidis. The cross rectangle intersection problem. *Computer Journal*, 38(3), 1995.
26. D. Kimber, Q. Liu, J. Foote, and L. Wilcox. Capturing and presenting shared multi-resolution video. In *SPIE ITCOM 2002. Proceeding of SPIE, Boston*, volume 4862, pages 261–271, Jul. 2002.
27. P. Ku, K.T. Winther, H.F. Stephanou, and R. Safaric. Distributed control system for an active surface device. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2001.
28. Q. Liu, D. Kimber, J. Foote, L. Wilcox, and J. Boreczky. Flyspec: A multi-user video camera system with hybrid human and automatic control. In *ACM Multimedia 2002, France*, Dec. 2002.
29. M. McDonald, D. Small, C. Graves, and D. Cannon. Virtual collaborative control to improve intelligent robotic system efficiency and quality. In *IEEE International Conference on Robotics and Automation*, April 1997.
30. N. Megiddo and K.J. Supowit. on the complexity of some common geometric location problems. *SIAM J. Comput.*, 13, 1984.
31. H. Moon and J. Luntz. Computing equilibria on superpositions of logarithmic-radial potential fields. In *The Workshop on Algorithmic Foundations of Robotics, December*, Dec. 2002.
32. D.M. Mount, R. Silverman, and A.Y. Wu. On the area of overlap of translated polygons. *Computer Vision and Image Understanding*, 64:353–61, 1996.
33. E. Mumolo, M. Nolich, and G. Vercelli. Algorithms and architectures for acoustic localization based on microarray in service robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2000.
34. S.G. Nash and A. Sofer, editors. *Linear and Nonlinear Programming*. The McGraw-Hill Companies, Inc, 1996.
35. L.E. Parker. Distributed algorithms for multi-robot observation of multiple moving targets. *Autonomous Robots*, 12(3), 2002.
36. B. Rogoff, E. Matusov, and C. White. *Models of teaching and learning: Participation in a community of learners*. Oxford, England: Blackwell, 1996.
37. R. Safaric, M. Debevc, R. Parkin, and S. Uran. Telerobotics experiments via internet. *IEEE Transactions on Industrial Electronics*, 48(2):424–31, April 2001.
38. R. Sagawa, K. Nishino, M.D. Wheeler, and K. Ikeuchi. Parallel processing of range data merging. In *IEEE/RSJ International Conference on Intelligent Robots and System (IROS)*, 2001.
39. R. C. Veltkamp and M. Hagedoorn. Shape similarity measures, properties, and constructions. In *Advances in Visual Information Systems, 4th International Conference, VISUAL 2000, Lyon, France, November 2-4, 2000, Proceedings VISUAL*, volume 1929, pages 467–476. Springer, 2000.