Coarse, Inexpensive, Infrared Tracking for Wearable Computing

Drexel Hallaway [*] Tobias Höllerer [†] Steven Feiner [*]

[*] *Department of Computer Science*
Columbia University
New York, NY 10027
+1 212 939 7000
{drexel, feiner}@cs.columbia.edu

[†] *Department of Computer Science*
University of California, Santa Barbara
Santa Barbara, CA 93106-5110
+1 805 893 8759
holl@cs.ucsb.edu

Abstract

We present a novel, inexpensive, coarse tracking system that determines a person's approximate 2D location and 1D head orientation in an indoor environment. While this coarse tracking cannot support precise registration of overlaid material, it can be used to drive user interfaces that can adapt to the quality of tracking available.

Our approach uses a set of strong infrared beacons, each of which broadcasts a unique ID. The beacons are deployed in the environment such that their zones of influence strategically overlap, partitioning the area of coverage into a set of uniquely identifiable fragments. We use a compound, omnidirectional infrared receiver, composed of a set of individual, directional infrared receivers, to infer 2D position (parallel to the ground plane) and 1D orientation (azimuth), employing a Kalman-filter–based architecture for smoothing and data integration with other tracking systems available. To test our ideas, we have applied them to a prototype head tracker, and present results from our tests.

1. Introduction

Augmented reality [3] is a potentially promising user interface metaphor for mobile information systems, offering the ability to spatially register relevant virtual information with the user's experience of the physical world. Much augmented reality research has concentrated on the design and use of relatively precise tracking technologies. These systems are typically limited in the size of the area that they track and the number of simultaneously tracked objects that they support, and are often relatively expensive. We are interested in how such precise tracking technologies might be complemented by coarser technologies that could significantly increase the area being tracked, at a modest increase in cost, and for a larger number of tracked objects.

To address this problem, we are developing an inexpensive, coarse, three-degree-of-freedom (3DOF), infrared-based tracking system. This system uses intersections and differences of the strategically overlapped zones of influence (ZOIs) of unsynchronized, world-stabilized infrared beacons, to provide a 2D position estimate relative to the ground plane. It also uses the world-frame beacon layout and the user-frame poses of its collection of wearable receivers to provide a 1D orientation estimate of user azimuth.

In the remainder of this paper, we first describe related work in Section 2. Next, in Section 3, we present our coarse, infrared tracker. Finally, we present our conclusions and plans for future work in Section 4.

Figure 1. *User wearing test helmet. Two beacons are visible on wall beyond.*

Proceedings of the Seventh IEEE International Symposium on Wearable Computers (ISWC'03)
1530-0811/03 $ 17.00 © 2003 IEEE

2. Previous Work

There is a significant body of research on tracking systems for large-scale indoor and outdoor environments. Hightower and Borriello [11] present a taxonomy of location systems for mobile-computing applications. Within their taxonomy, the positional aspect of our tracking method can be classified as a proximity-based infrared technology that yields either physical or symbolic location information in absolute coordinates, uses localized location computation, and does not provide recognition of tracked objects. The accuracy and precision of the system is variable and depends on the deployment scheme of the infrared transmitters in the environment.

High accuracy tracking has been achieved in research and commercial systems for up to room-sized areas using different technologies, such as magnetic [2, 18], hybrid inertial and ultrasonic [10], and infrared technologies [1, 22]. Most of these systems are tethered, but there are mobile wireless options available for some trackers [13].

Covering large parts of a building with these technologies can be quite expensive. Related research explores the tradeoff between cost and accuracy for such wide-area (multiple-room) indoor tracking. The most prominent technologies used for this purpose are ultrasound, IEEE 802.11b radio frequency (RF), dead reckoning, and infrared.

The Cricket [17] uses concurrent radio and ultrasonic signals to infer distance of sensors to beacons placed in the environment, achieving portion-of-a-room granularity. Randell and Muller [19] describe a similar approach, using four ultrasonic transmitters per room with reported tracking accuracies of 10–25cm. The Active Bat system [16] also uses ultrasound time-of-flight, but employs emitters in the mobile sensors that communicate with a grid of ceiling-mounted receivers. It has been shown to be effective, not only in position-tracking, but also in coarse orientation-tracking—especially when fused with superior local sensors for the latter.

Several research systems determine a person's location from signal quality measures of IEEE 802.11b (Wi-Fi) wireless networking. The RADAR system uses multilateration and pre-computed signal strength maps for this purpose [4], while Castro et al. [7] employ a Bayesian networks approach. At least one commercial venture is already marketing such services [8].

Dead-reckoning tracking approaches have been explored by [5] and [12]. Infrared (IR) is an attractive technology for location aware computing, since many mobile devices, such as palmtop computers, come with built-in IR ports, or can easily be IR enabled. In the Swarm of Locusts [20], infrared beacon cells provide coarse location and/or object tagging. Butz et al. [6] deploy strong infrared senders throughout a building, which broadcast either ID tags or contextual information to infrared-equipped clients, thereby enabling coarse location awareness—the receipt of a particular signal means simple proximity to an entity of interest.

We present here an experimental infrared tracker that also uses infrared beacons, but which exploits layout designs to create overlapping signals and a finer space partition, enabling our receiver and algorithms to infer more precise and continuous position and orientation estimates. Unlike other approaches, we use inexpensive, uncorrelated beacons, supporting an arbitrary number of tracked users.

3. Tracking Strategy

3.1 A Coarse Infrared Tracker

Our infrared-based tracking method uses a set of world-stabilized infrared beacons, and an array of mobile infrared receivers for each user. For beacons, we currently use battery-operated wireless Eyeled GmbH ELT-400 infrared transmitters [9]. Each one is user-configured to broadcast a unique numeric ID twice per second at 2400 baud. Butz and colleagues originally developed these transmitters for use in an architecture in which each beacon is mapped uniquely to a single entity—typically positioned nearby [6]. In their system, beacons are set up such that at most one beacon influences a given point in space, and the receipt of its signal by a hand-held computer with an infrared port means that the user is near that beacon's entity. This model is more logical than spatial. Beacon zones either must not ambiguously overlap, or ones that do must share the same semantics: two or more beacons broadcasting the same ID might be positioned near one another to provide a wider area of influence for the logical entity to which they map.

In contrast to this approach, we design and develop algorithms for beacon layouts with ZOIs (zones of influence) that intentionally overlap, so that the area of coverage is partitioned as uniformly as possible, given the coverage area, its shape, and the number of beacons currently available. Our approach also combines multiple IR receivers to increase accuracy and reliability.

Hardware and Setup Considerations. The mobile side of our infrared tracking system "watches" for beacon signals with a set of Extended Systems XTNDAccess Serial-to-IRDA infrared "dongle" receivers connected to a backpack-mounted laptop computer with Socket Communications PCMCIA-to-DB9 RS-232 adapters. In our work thus far, we have modeled each dongle as a point receiver, and have mounted eight of them on a helmet at 45° increments around a plane just above the user's head (Figures 1 and 2).

In modeling the characteristics of the beacons, our tests showed that the signal intensity of each beacon dimin-

Figure 2. *Dongle array on test helmet*

ished continuously as the signal was measured farther from the beacon's central axis at constant range. We first plotted a rough curve, accumulating a set of points, at each of which the receivers lost signal, even when (optimally) pointed directly at the beacon. This began our search for a best-case beacon ZOI in our model. This plotted region was more or less elliptical, with the beacon at one end of the major axis, a finding confirmed by the vendor's documentation.

Our initial plot provided a rough equipotential curve, at each point of which approximately the same signal energy was present—the minimum signal required to excite our receivers. We used the inverse-square law to back out a sense of what level of signal was propagating from the beacon along the various vectors to those plotted points, vectors described in 2D as angles measured from the beacon axis. We suspected that signal "falloff" might be Gaussian, since, despite their individual characteristics, several LEDs are clustered inside each beacon. In comparing the curves of best fit generated by models using Gaussian, geometric, and arithmetic falloff assumptions, it was clear that the Gaussian model was the best choice, and one surprisingly close to our observations. Figure 3 shows the Gaussian-model-driven curve which best fit our observations—a zone whose bounding box would be roughly seven meters long and 2.8 meters wide.

The primary, if slight, divergence of this Gaussian model from our observations was seen very near the beacon. Whereas the Gaussian model is somewhat "pointed" there, our observations implied a shape that was slightly more broadly curved than at the opposite end. We attribute that difference to what we assume are reflections internal to the beacon housings. While presumably very off-axis and resultantly weak, they would have a softening, fill effect on the shape. Given that this effect only makes the ZOI more elliptical, that we typically position beacons where users will not get very close to them, and that we were designing our system for fairly coarse-grained tracking, we determined that it was sufficient—for the positioning "side" of the system—to model the beacon ZOIs as 2D elliptical projections like those shown in the layout images of Figure 4. Each such ellipse represents a slightly simplified model of the intersection of a horizontal plane, at the height of the typical walking user, with the 3D ellipsoidinal ZOI of a beacon firing more or less horizontally, at or just above the level of the user's head. Given our coarse tracking expectations for this device, such simplifying assumptions have not seemed problematic as yet.

On the mobile, user-stabilized side of the system, we found an analogous situation. A dongle's ability to receive any constant-strength signal decreases the farther its pose is rotated away from the dongle-to-beacon vector. This means that the nearer a dongle is to a beacon, especially along the beacon axis, where the signal is stronger at any given range, the more it can be oriented away from the vector joining the two. The farther away it is, especially off the beacon's axis, where the signal is comparatively weaker, the more closely it must be pointed toward the beacon to receive the signal. This observation becomes useful later, as we describe the azimuth inference techniques: it means that distant, off-beacon-axis readings will have higher angular certainty than closer, on-axis readings. We also model this off-axis signal attenuation as Gaussian, as evidenced in the beacon-side signal-strength algorithm and the dongle-side azimuth-inference algorithms of Figures 5 and 6.

The setup approach we currently follow is to hand-define beacon locations and orientations, considering several strategic elements. Obviously, the space we wish to track must be covered. As we incrementally add elliptical ZOIs to the tentative plan, we monitor the largest few area fragments defined by the intersecting ellipses, trying to

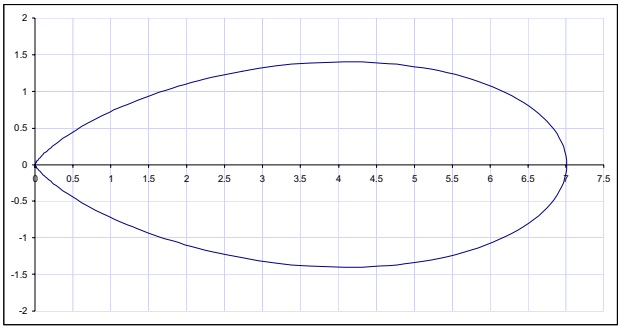

Figure 3. *Bounding our infrared beacon's 2D ZOI, this is the curve of best fit (measured in meters), based on a Gaussian model of intensity falloff away from the "mean" central axis. Beacon is at the origin, pointing right.*

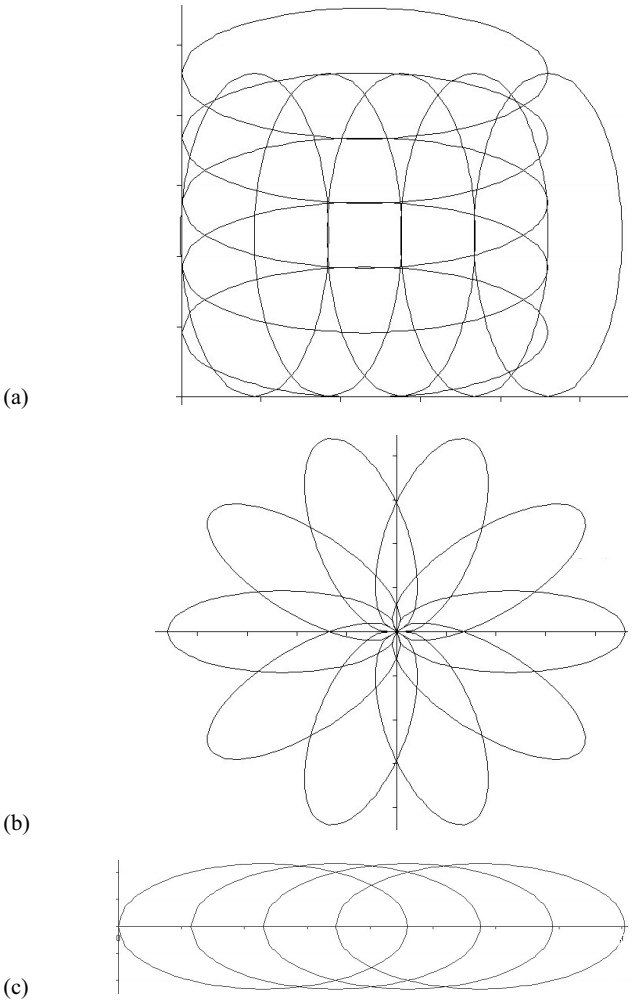

Figure 4. *Efficient layouts for: (a) square room or section; (b) round room with finer detail toward center; (c) hallway or long, narrow room.*

position the new ZOIs to cut them, continually reducing the size of the largest elliptical fragment in the layout. Furthermore, while we have encountered few problems with this, we try to avoid choosing any beacon pose that would be likely to cause a system-confusing signal reflection off an environmentally immobile object, such as a wall. Finally, although we have not exploited it in our work here, we observe that in certain environments, more or less linear pathways between immobile objects seem highly probable user trajectories. Given such a segment, if other constraints allow, tracking benefits further accrue from the parallel alignment of a pair of beacons, creating a long elliptical intersection that overlays such a potentially high-traffic segment.

Once all the design decisions are made, we store the beacon-pose ZOI layout in a configuration file. Figure 4 shows several layout styles we have considered. We currently use the 10-beacon, orthogonal layout of (a) in our laboratory and in the test results we later show.

Algorithms and Software Architecture. At the lowest level in our architecture, there is one dongle driver for each dongle. Each ID signal received by a particular dongle from any beacon is an event that calls higher-level updating logic on the orientation side of the tracker, and time-stamps and caches the received ID for subsequent batch use on the position side.

On the orientation side, data resources include the positions and orientations for each IR beacon, which we encapsulate, along with signal-intensity-computing logic, in an object called *BeaconProfile*. Also key are the positions and orientations of the dongle receivers in the user's mobile reference frame. These are stored within a class we call *DonglePose*, which also contains methods that support the user-azimuth inference logic.

When a particular dongle and its driver receive a beacon's ID signal, that event immediately invokes logic in its dedicated, higher-level *IrDAStation* construct. It retrieves from the single, yet-higher-level *IrDADriver* the particular *BeaconProfile* that matches that ID, and gets the most recent user position estimate as well. The *BeaconProfile* and user position are passed to *IrDAStation*'s dedicated *DonglePose*, which in turn calls a method on the *BeaconProfile* it was passed, which finally returns an estimate of beacon signal strength, based on the user position estimate it was given. *DonglePose* then computes an estimate of user azimuth and its angular uncertainty, based on the position estimate and the beacon signal strength. Pseudocode for some of this logic is shown in Figures 5 and 6. Finally, *IrDAStation* passes the azimuth estimate and variance to a dedicated Kalman filter, which handles each such event on a single-constraint-at-a-time (SCAAT) basis [21], and caches the time-stamped ID for near-term batch use by the position-inference logic. Dynamic, graphical display of this user azimuth estimate is shown in Figure 12 as the short black line emerging like the hand on a clock from the white estimate dot.

Above the dongle driver sits a higher-level, position-only driver that frequently checks these cached IDs and their time-stamps, assembling them into a set of beacon IDs that includes both those received since its last iteration, and those whose time-stamps are recent enough (≤500 ms) that it is rather likely that the reason they were not received since the last iteration is that their beacons were merely in between their 2Hz bursts. Given a working set of IDs that the driver believes have been received or are likely receivable, space-partitioning and lookup-facilities are invoked in a construct called the *AreaCollection*, submitting the ID set for its processing. Based on the one-time start-up and the runtime lookup algorithms described below, *AreaCollection* can retrieve any fragment in constant time.

```
beacon_prof      // BeaconProfile passed in call
userPos          // user's world position passed in call
dongle_var       // dongle attenuation variance
user_to_dongle   // constant angular pose in user frame

user_beacon_range ← beacon_prof.getRange( userPos )
user_beacon_angle ← beacon_prof.worldAngleFrom( user-
   Pos )
max_off_axis ←
   sqrt[ 2 * dongle_var * ln( beacon_prof.getIntensity( ) ) ]
lower_user_rot ← user_beacon_angle
   – ( user_to_dongle + max_off_axis )
higher_user_rot ← user_beacon_angle
   – ( user_to_dongle – max_off_axis )
half_rot_range ← ( higher_user_rot – lower_user_rot ) / 2
azimuth_estimate ← lower_user_rot + half_rot_range
azimuth_std_dev ← half_rot_range / 2
   // assuming range is 2 std. devs. off axis
```

Figure 5. *DonglePose's angle and variance algorithm, simplified when all dongles are at the user origin—the general case is more complex.*

```
userPos          // user's world position passed in call
sig_variance     // constant variance of signal over angle
min_received     // constant minimum receivable signal

userPosBF ← transformToBeaconFrame( userPos )
th ← arctan( userPosBF.y / userPosBF.x )
range_squared ← userPosBF.y * userPosBF.y
   + userPosBF.x * userPoBFs.x
signal_at_1 ← e ^ ( –th * th / ( 2 * sig_variance ) )
   / sqrt( 2 * PI sig_variance )
signal_at_point ← signal_at_1 / range_squared
return signal_at_point / min_received
```

Figure 6. *BeaconProfile.getIntensity algorithm.*

Recall that the strategy is to exploit the overlaps of these strategically laid-out elliptical zones to create a partition of the area of coverage. We begin with an empty "universe" of coverage: some arbitrary, target shape. When the first elliptical ZOI, e_1 is added, it partitions the universe into a fragment inside e_1, which is e_1, and the remaining universe fragment outside e_1. At this point e_1 is one of two fragments in the space partition, yet it remains a complete ellipse. As any subsequent ellipse e_n is incrementally added to the system model, each fragment f—in the partition as it existed before the addition—whose intersection with e_n is non-empty (including the remainder of the universe) is partitioned into sub-fragment f_{en} inside e_n and $f_{\sim en}$ outside it. As a side effect, e_n is itself partitioned into the set containing all of its intersections with all previous fragments f, and any remaining portion outside all such fragments—the latter of which is, interestingly, its intersection with the previously remaining universe fragment. After any number of such incremental additions, the space remains a partition, each area fragment of which is uniquely defined by the set of elliptical ZOIs it is inside, and the set of ellipses it is outside. As Figure 7 illustrates, a binary number with the same number of digits as there are ZOIs, suffices to encode such a unique fragment identification—each digit maps to a ZOI, zero means "out" and one means "in."

In the abstract, ignoring layout-dependent impossibilities, the set of all possible beacon combinations, given n beacons, is of cardinality 2^n. When applied to a particular layout domain, each combination maps to the area fragment in which a point-modeled receiver would have to reside to receive that combination of signals. Obviously, many such combinations map to area-fragments that are empty—in the modeled layout, there is no region in which that particular combination of beacons received and not received could occur. Frequently, such combinations map to non-singular regions, not especially helpful for position tracking. Is it realistic to assume that we will never encounter empty-fragment combinations? And, are non-singular fragments the best choice?

Relaxing one of these constraints—that of subtracting ZOIs serviced by beacons whose IDs were not received—results in another set of fragments. While this set is not a partition (they overlap one another), these simple intersections produce the fragments in which a set of IDs that were received could have been. Each of these fragments has several attractive properties: (1) it is always singular; (2) it is always a superset of its corresponding subtraction-enforcing fragment; and (3) it is less often empty. Many combinations, mapping to empty fragments in the subtrac-

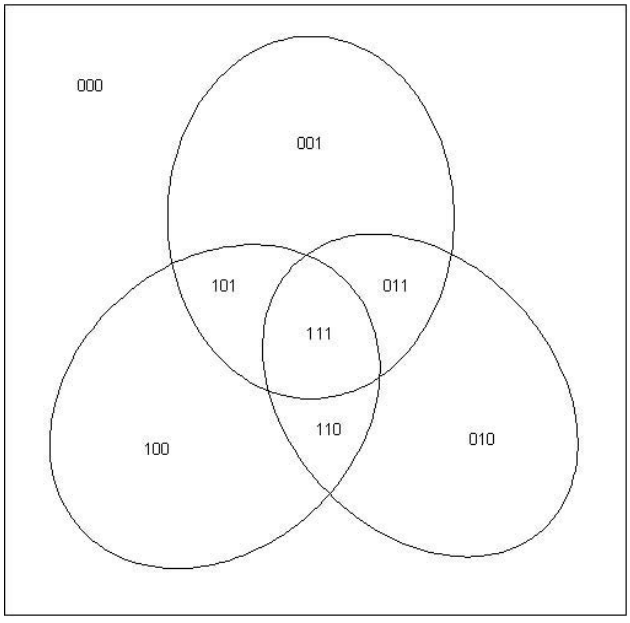

Figure 7. *ZOI fragment binary encoding strategy.*

tion-enforcing partition, map to non-empty ones in the simple-intersection set. And, no combination that maps to an empty fragment in the simple-intersection set will map to a non-empty one in the subtraction-enforcing partition—simple intersections are just more "productive" of usable fragments.

In answer to the question posed above, it is our sense, given environmental and dynamic factors we could never model, that we are more confident that we should have received what we did, than we are certain that we should not have received what we did not. For this reason, among others, we also precompute the set of simple-intersection fragments during initialization, ensuring that the set can be indexed identically to the subtraction-enforcing fragment partition.

In practice, the abovementioned *AreaCollection* is initialized from the configuration file in which we stored the layout design. Each fragment—both the simple-intersection and the subtraction-enforcing versions—is pre-computed at startup. As in Figure 7, the "universe" fragment is accorded the internal ID of zero—it is "out" of all the ZOIs. Whatever logical ID the layout designer chose to assign to each ZOI, the incremental initialization routine maps it to an internal ID, the smallest, yet-unused power of two. This fast, but tedious, runtime mapping could be avoided by simply choosing logical IDs that were powers of 2, and initializing them in order. After initialization, each fragment can be identified and located, bitwise uniquely, by the internal ID generated by the bitwise OR of all the power-of-two internal IDs of each ZOI it is in—zero bits for the ones it is outside.

Pseudocode for the initialization algorithm, which sets up data structures supporting constant-time fragment lookups, is presented in Figure 8. In that figure, *Fragments* is a vector of subtraction-enforcing fragments, and *Intersections* is a vector of simple intersections. Looking up a fragment, given a set of logical IDs, simply involves a fast lookup of the corresponding internal IDs (a step that could be omitted as noted above), and then retrieval can be accomplished using the simple algorithm of Figure 9. Given a small number of beacons (we currently use 10), and the fact that the arrays used in Figures 8 and 9 contain references, not large memory allocations, for clarity and elegance we currently forgo the space efficiencies hash tables might offer over sparsely populated arrays.

What if the simple-intersection fragment is also empty? In our implementation, an empty fragment is a non-update—the system maintains the status quo until it gets a meaningful change. But, what might generate such a condition, and should we be concerned? An empty simple-intersection fragment means that the set of IDs that the system received during its sliding time window, maps to a set of ZOIs, at least one of which we have modeled as being disjoint from the rest of the set. Given the sliding time window in which we accumulate and retain beacon "hits," user motion at a high speed might allow an ID to stay in the working set for at most a half second longer than theoretically ideal. This can cause momentary situations in which the user is not simultaneously inside all the ZOIs mapped to by this sometimes-behind-the-times working set of beacon IDs.

Aside from the above consideration, several other possibilities are:

```
Ellipses        // an input vector of whole, elliptical zones
Intersections   // a vector of size 2 ^ Ellipses.length
Fragments       // a vector of size 2 ^ Ellipses.length

Fragments[ 0 ] ← universe
Intersections[ 0 ] ← universe
newID ← 1
m ← 0           // indexes vector of elliptical zones
while m < Ellipses.length
   newEllipse ← Ellipses[ m ]
   i ← 0
   while i < newID
      tempFrag ← Fragments[ i ]
      Fragments[ i ] ←
         Fragments[ i ] SUBTRACT newEllipse
      Fragments [ i + newID ] ←
         tempFrag INTERSECT newEllipse
      Intersections[ i + newID ] ←
         Intersections[ i ] INTERSECT newEllipse
   repeat
   newID ← newID * 2
   m ← m + 1
repeat
```

Figure 8. *Initialization algorithm for infrared beacon ZOIs.*

```
IIDs            // array of internal, power-of-two beacon IDs
Fragments       // array of ellipse and universe fragments
WholeEllipses   // array of whole ellipses cached above

j ← 0
fragIndex ← 0
while j < IIDs.length
   fragIndex ← fragIndex OR IIDs[ j ]
repeat
if AND( NOT_EMPTY( Fragments[ fragIndex ] )
            SINGULAR(Fragments[ fragIndex ])
            NOT_TINY(Fragments[ fragIndex ])
      ) return Fragments[ FragIndex ]
else return Intersections[ FragIndex ]
```

Figure 9. *Lookup algorithm for area-zone fragments, under the policy of usually using the knowledge about beacons not received.*

(1) an offending beacon signal has bounced off some reflective surface into the area surrounding our user—an area in which we had not modeled it as receivable;
(2) a beacon's physical position doesn't match its state in the configuration file;
(3) a beacon is broadcasting an ID that doesn't match the one assigned to it in the configuration file.

Problems 2 and 3 are simple, human configuration or setup errors, which we assume would be detected and corrected during post-setup tests. Problem 1, however, requires further thought. The layout schemes we have considered up to this point, some of which are pictured in Figure 4, are ones in which the likelihood of such a bounce is extremely low, given:

(1) beacons pointed away from walls;
(2) beacon signals decaying just as or before they hit any opposite walls, or with any bounce zone a subset of the already-modeled ZOI.

Pose choices can usually avoid the likelihood of detrimental bounces. In a room whose dimensions are significantly smaller than the longer axis of the beacon ellipse as modeled, the beacons' developers mention that it is possible to bend the diodes outward from the central axis, thus altering the shape and length of a beacon's receivable volume [personal communication]. Doing so would require each beacon to have its own, carefully calibrated model. In our lab tests, in which the layout is that of Figure 4(a), without any of these avoidance techniques, we have experienced no ill effects we could attribute to such bounces. We further note that even forced bounces, as long as they are constant, can be handled as follows. If a wall, for instance, cuts across a necessary beacon's ZOI, then at initialization, or even in configuration, that ZOI can be broken down into two portions: that covered before the signal hits the wall, and that covered by the signal's reflection off the wall. The working ZOI would then be the union of these two portions.

That said, random reflections of IR signal by moving surfaces cannot be modeled. We observe, though, that it would take more than a flicker of light to confuse the system. Rather, four bytes of a sporadic, 2Hz, 2400-baud signal would have to be legible in an unexpected area.

In Figures 10 and 12, we present screen-shots of our test program at the end of some walk-arounds in our lab tracked by this infrared system. In the upper image of Figure 10, for example, the *Intersections* area is the lighter-shaded, larger fragment, bounded top and bottom by the third horizontal ellipse (from the top). The *Fragments* area is the darker-shaded, central subset of that—the wedge bounded by the second and fourth horizontal ellipses. The later-discussed ellipse of confidence appears as a full, shaded ellipse with a white dot at its centroid.

Figure 10. *Casual user walk-arounds tracked by the infrared system in our lab.*

3.2 Filtering the Raw Tracker Data

Once a coherent area fragment is returned from the area collection, what happens? One policy we have investigated is that of using the centroid of the fragment's axially aligned bounding box as the 2D position meas-

urement. Testing shows that those centroids are sometimes farther away from the user's last known position than seems helpful, and that using the centroid as the measurement is excessive, especially with the largest of the simple-intersection fragments. To address this issue, we have also investigated another strategy. We evaluate the measurement differential (from the last update) in terms of implied velocity. If that velocity exceeds a configurable maximum-velocity assumption (2 m/sec in our walk-around tests), the differential is scaled back appropriately, and added to the last position estimate. The measurement updates are always in the "right direction," but are never far enough in that direction to imply a velocity above the configurable cap. Second, we handle many kinds of uncertainty by using a Kalman filter [15], and some of its output is employed to further constrain the measurement fragment.

Apart from other techniques, raw measurements based on the above layouts and algorithms could be very noisy. For much of the solution to this problem, we turned to the Kalman filter. We use the centroids of the area fragments returned by the *AreaCollection* as the *x* and *y* sensor-measurement inputs to the Kalman filter—except when we attenuate large changes with our configurable velocity-cap assumption. The width and height of the axially aligned bounding box for that fragment provide the basis for estimating the measurement's standard deviation—also a necessary input into the Kalman filter.

We employ a second Kalman filter for the user azimuth estimates that flow immediately from each ID read. That azimuth is represented as an Euler angle, user yaw.

Back on the position side of the device, we also benefit from the Kalman filter output by using it for pre-filtering feedback on the next cycle. We cache a representation of what we call an "ellipse of confidence" around the current filter estimate. Graphical examples of this appear in the images of Figures 10 and 12: it is always a full, shaded ellipse (often the only one)—with a white dot representing the estimate at its centroid. Because of what is filtered, this ellipse is axially aligned with the 2D coordinate frame, and the height and width of its bounding box are proportional to the standard deviations we get from the filter.

We use this ellipse for more than graphical output, however. It also has a role in smoothing noisy data. Currently, we intersect its most current version with the next area fragment output by the *AreaCollection*, and pass to the filter a measurement we derive from that intersection, rather than just the raw fragment itself. Since we believe the user to be within the filter's estimate-confidence bounds, and since the fragment obtained from the *AreaCollection* is also very likely to contain our mobile receivers, the most likely subset of both would seem to be their intersection.

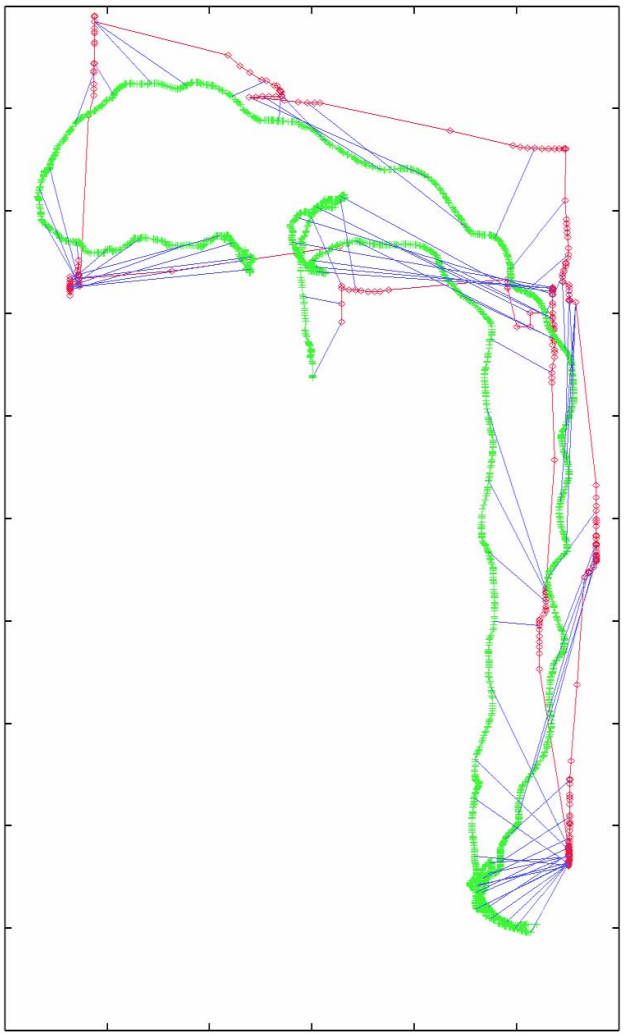

Figure 11. *IR tracker trajectory (lighter, more linear, circle nodes) overlaying the InterSense IS-900 (heavier, curved). This is an area roughly 1/3 the width of the lab area in Figure 10 (half-meter ticks), so deviations are exaggerated. Line segments show temporal matches between the two at one-second intervals.*

3.3 Evaluating a Coarse Tracker's Resolution

Figure 11 presents a typical example of the IR tracker's output overlaying that of the InterSense IS-900 [13] ceiling tracker, which we used for "ground truth." It should be noted that this image is scaled to show an area roughly 1/3 the width of those in Figure 10, so deviations are exaggerated here. At one-second intervals we have provided leader lines between simultaneous estimates from the ceiling tracker and the IR tracker. Occasional "fans" of these leader lines, connecting closely packed IR estimates to the more accurate IS-900 curve, serve to illustrate the "stall-and-catch-up" positional nature of the IR device as it now stands.

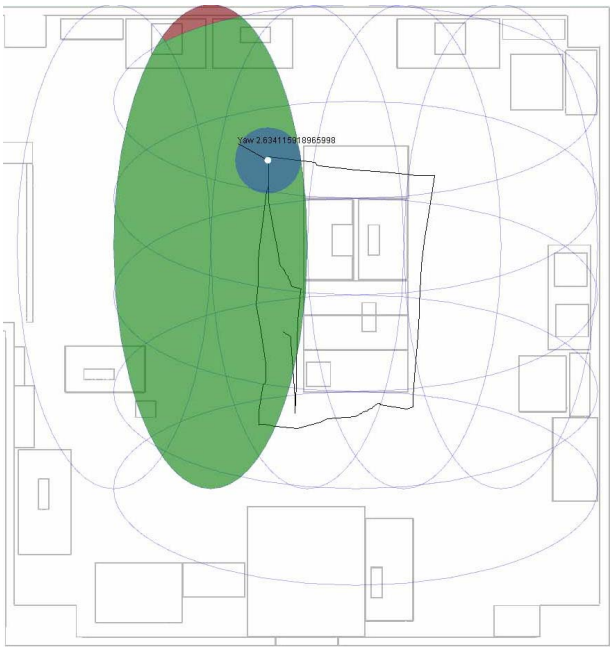

Figure 12. *User's azimuth indicator and angle number. Note: this walk-around was generated with different filter settings—generating higher latency, but a more stable path image.*

When multiple beacons are in range, typical latency in position tracking is on the order of a second or so. Worst-case lags, typically driven by missed beacon broadcasts in this crude test-bed implementation, were occasionally as much as a few seconds. Even under the most pessimistic view, this seems usable in the context of a user doing a stroll-and-stop browse around a museum or conference floor, for example. Azimuth latency is often less than a second, but never worse than worst-case positional experiences. We attribute the lower latency to our use of a SCAAT [21] filtering approach for azimuth.

A key cause of latency beyond our current control is the slow, 2Hz beacon broadcast rate. During rapid movements, the user might pass through a narrow section of a ZOI without reading its signal at all. Rapid user rotation might also introduce delay, if none of the dongles get all four bytes of a beacon's broadcast—generating a miss in the current structure.

Positional accuracy, when the user pauses at some spot, is typically on the order of a meter, given the density and uniformity characteristics of the relatively sparse layout design we test here. Layouts populated more densely with beacons will enjoy a finer granularity of positional precision. In the worst case, even if only one beacon is received, positional error is bounded by the dimensions of its ZOI. More typically, the error is bounded by the dimensions of the simple-intersection fragment implied by the multiple beacon IDs currently receivable. Azimuth accuracy is on the order of 5–10° in the typical, settled case. Rapid user rotations are, of course, not immediately reflected, given the latency, but slow steady turns are often registered rather smoothly. This tracking strategy seems, if anything, more stable in orientation than it does in position: positioning requires the reliability of a set of IDs for inferencing. Azimuth, given a rough position estimate, only requires a single beacon "hit" for a reasonable update.

4. Conclusions and Future Work

We have described our early experiences with a coarse infrared-beacon tracker we are developing. The device estimates 2D position from the set of infrared signals it receives—more precisely, from the spatial inferences it implicitly makes over the set of modeled zones to which that signal set maps. Based on the current position estimate and models of receiver poses and beacon locations, it also infers user azimuth from individual beacon "hits." We have been pleasantly surprised, in the context of modest, coarse-tracking expectations, at how well this device performs.

The Kalman filter we employ for position tracking is being applied to an atypical domain—one in which some of its assumptions arguably do not hold. Kalman filtering assumes that the probability distribution of measurements is normal (Gaussian). One can reasonably assert that having received signal set *S*, the probability of being in, say, the square decimeter of the fragment farthest from the operative beacons, is not equal to—indeed is surely quite a lot less than—the probability of being in the nearest one. If so, the probability distribution of the reception-location across these elliptical ZOIs, or indeed their fragments, is certainly not Gaussian. That the filter performs as well as it does, in our view, merely serves to highlight the essentially forgiving nature of Kalman's algorithm—another example of the benefits of applying it where some of its theoretical assumptions may not hold. That said, we are interested in investigating other alternatives to simple Kalman filtering, where the underlying assumptions may not differ as much from the physical facts.

It is our reasonable intuition that the accuracy of our tracker is a function of the density of the beacon distribution. We would like to do performance testing with several layouts, and find a sound means of expressing the accuracy level that can be expected from this device, given a particular layout scheme.

This tracking approach would greatly benefit from increasing the frequency of the beacon broadcasts. Were we to make or acquire beacons that could broadcast at, say, 10Hz, instead of the current 2Hz, there would be two improvements. The average time would be reduced between when a user entered a ZOI, and when its beacon broadcast was received. Also reduced would be the time

window during which we buffer and use beacon IDs received—assuming the user remains in those ZOIs. One risk of increasing frequency, however, is rooted in the reality that the current system uses simple, autonomous beacons, which receive no communication from the rest of the system. Internal clocks presumably drift with respect to the other beacons, so one would expect situations in which one beacon's broadcast "stomped" another's, causing both signals to be lost. Expensive, high-precision systems employ beacon synchronization to support time-slicing for higher frequencies. Doing such with this system would increase its cost and complexity significantly.

We can imagine replacing the eight dongles we used here by designing a smaller, custom receiver that would house many more uniquely posed diodes.

Furthermore, we have considered introducing some vertical displacements into the beacon layouts—especially some beacons firing down from the ceiling. Between these two enhancements, we would anticipate being able to provide something more like 3DOF orientation tracking, and certainly coarse 3DOF position tracking, as well. At minimum, we need to extend the layout algorithms and space representation to support the notion of beacons that do not fire in a more or less horizontal manner.

We look forward to using this device in hybrid combination with other inexpensive sensors, to both improve accuracy and provide more degrees of freedom in tracking estimates. Using an inclinometer might extend the 1DOF azimuth value to a 3DOF orientation one within certain ranges. Adding an altimeter would track vertical motion in elevators or staircases [14].

5. Acknowledgments

The research described here is funded in part by ONR Contracts N00014-99-1-0249 and N00014-99-1-0394, NSF Grants IIS-00-82961 and IIS-01-21239, and a gift from Microsoft.

6. References

[1] 3rdTech Corp., http://www.3rdtech.com/HiBall.htm, 2001.

[2] Ascension Technology Corp., http://www.ascension-tech.com.

[3] Azuma, R. T., "A Survey of Augmented Reality," *Presence: Teleoperators and Virtual Environments*, vol. 6(4), pp. 355–385, 1997.

[4] Bahl, P. and V. N. Padmanabhan, "RADAR: An In-Building RF-based User Location and Tracking System," *Proc. InfoCom 2000 (Joint Conf. of the IEEE Computer and Communications Societies)*, vol. 2, 2000, pp. 775–784.

[5] Borenstein, J., H. Everett, L. Feng, and D. Wehe, "Mobile Robot Positioning: Sensors and Techniques," *Journal of Robotic Systems*, vol. 14(4), pp. 231–249, 1997.

[6] Butz, A., J. Baus, A. Krüger, and M. Lohse, "A Hybrid Indoor Navigation System," *Proc. IUI 2001 (Int'l Conf. on Intelligent User Interfaces)*, Santa Fe, NM, 2001, pp. 25–32.

[7] Castro, P., P. Chiu, T. Kremenek, and R. R. Muntz, "A Probabilistic Room Location Service for Wireless Networked Environments," *Proc. UbiComp 2001 (Int'l Conf. on Ubiquitous Computing)*, Atlanta, GA, 2001, pp. 18–34.

[8] Ekahau, Inc., Accurate Positioning in Wireless Networks, Ekahau Positioning Engine 2.0, http://www.ekahau.com.

[9] Eyeled GmbH, http://www.eyeled.com/en/1/1.html, 2001.

[10] Foxlin, E., M. Harrington, and G. Pfeifer, "Constellation: A Wide-range Wireless Motion-tracking System for Augmented Reality and Virtual Set Applications," *Proc. SIGGRAPH '98 (ACM Conf. on Computer Graphics and Interactive Techniques)*, 1998, pp. 371–378.

[11] Hightower, J. and G. Borriello, "Location Systems for Ubiquitous Computing," *IEEE Computer*, vol. 34(8), pp. 57–66, 2001.

[12] Höllerer, T., D. Hallaway, N. Tinna, and S. Feiner, "Steps Toward Accommodating Variable Position Tracking Accuracy in a Mobile Augmented Reality System," *Proc. AIMS 2001 (Int'l Workshop on Artificial Intelligence in Mobile Systems)*, Seattle, WA, August 4, 2001, pp. 31–37.

[13] InterSense, Inc., IS-900 Wide Area Precision Motion Tracker, http://www.isense.com, 2001.

[14] Judd, C. T., "A Personal Dead Reckoning Module," *Institute of Navigation's ION GPS*, Kansas City, MO, September, 1997.

[15] Kalman, R. E., "A New Approach to Linear Filtering and Predictive Problems," *Trans. ASME—Journal of Basic Engineering*, vol. 82(Series D), pp. 35–45, 1960.

[16] Newman, J., D. Ingram, and A. Hopper, "Augmented Reality in a Wide Area Sentient Environment," *Proc. ISAR 2001 (IEEE and ACM Int'l Symp. on Augmented Reality)*, New York, NY, 2001, pp. 77–86.

[17] Priyantha, N. B., A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," *Proc. MobiCom 2000 (ACM Int'l Conf. on Mobile Computing and Networking)*, Boston, MA, 2000, pp. 32–43.

[18] Raab, F. H., E. B. Blood, T. O. Steiner, and H. R. Jones, "Magnetic Position and Orientation Tracking System," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 15(5), pp. 709–718, 1979.

[19] Randell, C. and H. Muller, "Low Cost Indoor Positioning System," *Proc. UbiComp 2001 (Conf. on Ubiquitous Computing)*, September, 2001, pp. 42–48.

[20] Starner, T., D. Kirsch, and S. Assefa, "The Locust Swarm: An Environmentally-Powered, Networkless Location and Messaging System," *Proc. ISWC '97 (IEEE Int'l Symp. on Wearable Computers)*, Cambridge, MA, October 13–14, 1997, pp. 169–170.

[21] Welch, G. and G. Bishop, "SCAAT: Incremental Tracking with Incomplete Information," *Proc. SIGGRAPH '97 (ACM Conf. on Computer Graphics & Interactive Techniques)*, Los Angeles, CA, August 3–8, 1997, pp. 333–344.

[22] Welch, G., G. Bishop, L. Vicci, S. Brumback, K. Keller, and D. Colucci, "The HiBall Tracker: High-Performance Wide-Area Tracking for Virtual and Augmented Environments," *Proc. VRST '99 (ACM Symp. on Virtual Reality Software and Technology)*, London, December 20–23, 1999, pp. 1–11.