Cappella: Establishing Multi-User Augmented Reality Sessions Using Inertial Estimates and Peer-to-Peer Ranging

John Miller  
Carnegie Mellon University  
jmiller4@andrew.cmu.edu

Elahe Soltanaghai  
University of Illinois at Urbana-Champaign  
elae@illinois.edu

Raewyn Duvall  
Carnegie Mellon University  
rduvall@andrew.cmu.edu

Jeff Chen  
Carnegie Mellon University  
kochengec@andrew.cmu.edu

Vikram Bhat  
Carnegie Mellon University  
vgbhat@andrew.cmu.edu

Nuno Pereira  
Carnegie Mellon University  
npereira@andrew.cmu.edu

Anthony Rowe  
Carnegie Mellon University  
agr@ece.cmu.edu

ABSTRACT

Current collaborative augmented reality (AR) systems establish a common localization coordinate frame among users by exchanging and comparing maps comprised of feature points. However, relative positioning through map sharing struggles in dynamic or feature-sparse environments. It also requires that users exchange identical regions of the map, which may not be possible if they are separated by walls or facing different directions. In this paper, we present Cappella\(^1\), an infrastructure-free 6-degrees-of-freedom (6DOF) positioning system for multi-user AR applications that uses motion estimates and range measurements between users to establish an accurate relative coordinate system. Cappella uses visual-inertial odometry (VIO) in conjunction with ultra-wideband (UWB) ranging radios to estimate the relative position of each device in an ad hoc manner. The system leverages a collaborative particle filtering formulation that operates on sporadic messages exchanged between nearby users. Unlike visual landmark sharing approaches, this allows for collaborative AR sessions even if users do not share the same field of view, or if the environment is too dynamic for feature matching to be reliable. We show that not only is it possible to perform collaborative positioning without infrastructure or global coordinates, but that our approach provides nearly the same level of accuracy as fixed infrastructure approaches for AR teaming applications.

Cappella consists of an open source UWB firmware and reference mobile phone application that can display the location of team members in real time using mobile AR. We evaluate Cappella across multiple buildings under a wide variety of conditions, including a contiguous 30,000 square foot region spanning multiple floors, and find that it achieves median geometric error in 3D of less than 1 meter.

1 INTRODUCTION

Driven by advances in visual-inertial odometry (VIO), simultaneous localization and mapping (SLAM), and miniaturized depth sensing technologies, we are seeing augmented reality (AR) become more accessible on a wide variety of platforms. Mobile phones are now equipped with dedicated hardware to enable richer AR experiences, including multiple cameras, specialized processors, ultra-wideband (UWB) ranging radios [3], and small LiDAR depth sensors. Navigating applications like Google Maps, utilities like IKEA Place, and games like Pokemon Go have shown some of the early potential for AR on mobile phones. These applications are typically built on top of existing AR frameworks like ARKit, ARCore, MixedReality Toolkit, and Vuforia.

Primarily driven by table-top gaming, we are now seeing applications where multiple users interact with shared content [2]. This is relatively straightforward in controlled environments, where all users can exchange a common set of reliable visual features. However, one could imagine extending these applications to larger and more complex domains, where simple feature sharing is infeasible. Take, for example, a first responder or firefighter application, where teams of users navigate through a previously unexplored or harsh (damaged/modified) environment while wearing an AR headset. With a robust multi-user AR platform, first responders could see the status and position of fellow teammates and the location of support vehicles even through walls without any a priori scene information. We already see this is a challenge in systems like the Army Integrated Visual Augmentation System (IVAS) [49] which is using modified Hololens 2 headsets for indoor/outdoor team awareness.

In the mobile phone context, this same type of platform could help find a friend at a concert venue that is both large and with highly dynamic lighting and dynamic staging.

Localization of users and other objects in the environment within a common coordinate system is a critical requirement for wide-area multi-user AR applications. In order to overlay virtual content from the user’s perspective that is “anchored” to the physical world, it is necessary to track the pose of the user’s display relative to the world. As the user moves and rotates the display, the projected content needs to move accordingly, which requires accurate 6DOF motion tracking. With a single user, it is sufficient to perform this tracking with respect to any arbitrary starting pose. However, the

\(^1\)Like its musical inspiration, Cappella utilizes collaboration among agents to forgo the need for instrumentation.

Figure 1: Cappella offers a distributed infrastructure-free relative positioning framework that allows multiple mobile users to create a collaborative AR session even in non-line of sight.
problem becomes more challenging with multiple users since each tracking instance must share the same 6DOF origin. This problem can be slightly simplified to 4DOF when the gravity direction for all devices is assumed to be known using inertial (IMU) sensors.

Current AR frameworks like Apple’s ARKit and Google’s ARCore provide multi-user support by sharing visual (and depth) features between users to establish their coordinate system. As each user detects distinguishable features in the environment, these features are collected into a map using SLAM. By sharing this map, other users can localize themselves if they detect the same visual features. However, obtaining a successful feature match requires the users to view the surrounding scene from a very similar perspective [2]. In addition, feature matching struggles if objects have been moved, become occluded, or if lighting conditions have changed. In many practical applications, users are often taking disjoint paths through the environment, so there will likely be no common visual features for map matching, either because the users are separated by walls, are facing different directions, or because the environment is too visually uniform. Additionally, in order to provide building-scale coverage and beyond, it is necessary to maintain a large and dense feature map, which quickly becomes impractical to store and share.

This paper addresses these challenges by proposing Cappella, a distributed relative positioning framework that allows multiple users to create an on-demand collaborative AR session. Cappella uses peer-to-peer distance measurements to establish a common coordinate frame, so it does not rely on any positioning infrastructure or sharing of map data. In our implementation, range measurements come from UWB radios, which are now available on the latest generation of mobile phones and specialized AR headsets [3]. To our knowledge, Cappella is the first multi-user AR system to use peer-to-peer ranging to establish a coordinate frame rather than external infrastructure or feature maps. Cappella’s key innovation is the design of a collaborative particle filter that jointly estimates the poses of all AR users relative to each other. Since it does not rely on sharing visual features, this approach is broadly applicable to static or dynamic environments, both indoor and outdoor, including scenarios where the need for visual pre-mapping is a nonstarter.

To achieve this, Cappella captures the local inertial information from each individual AR user, providing a 6DOF odometry estimate of this user over time. While tracking motion, Cappella collects distance ranges (using UWB) to other users and combines these information sources using a particle filter. Like most inertial tracking systems, VIO tracking estimates are smooth and locally accurate but drift over time and are only relative to the start pose. UWB ranges, on the other hand, provide absolute distance information and do not drift over time, but they are infrequent and noisy. By combining these complementary sensors, we achieve the best of both worlds. The absolute nature of UWB ranges allows us to correct VIO drift over time, while noise in UWB readings is smoothed by the VIO. In addition, the distributed architecture of Cappella allows each user to locally estimate the pose of other AR users with minimal message exchange between users.

One core challenge in implementing the joint particle filter is the state-space explosion as the number of AR users grows, a common dilemma faced by the robotics community in systems which track a large number of state variables. A common robotics solution is to use Rao-Blackwell factorization (RBPF) to reduce the required number of particles to a tractable level [59]. Whereas many RBPF implementations, such as the popular LiDAR SLAM package GMapping [25], perform factorization over a grid or landmark map, Cappella performs factorization over other users’ locations. This has a “collaborative” effect, wherein ranges to one user can improve the location estimates of the other users. Compared to a more traditional particle filter where each user is tracked independently, we show that the collaborative approach is able to improve accuracy while maintaining a reasonable memory footprint that grows linearly with the number of tracked nodes. In addition, our filter formulation allows for sporadic UWB and VIO updates, loosening communication constraints in the system design over methods that rely on fixed-rate updates.

In order to prototype Cappella in a teaming use-case, we developed a mobile AR application for iOS. Our technique is applicable to any relative tracking system that uses inertial data and ranging estimates and hence could also be applied to AR headsets in hands-free applications like aiding first responders. Since UWB APIs are not available to mobile phone developers at the time of this writing, we created peer-to-peer ranging firmware for the MDEK1001 evaluation module from Decawave. The firmware allows a phone to pair with the MDEK module over BLE, which discovers and ranges with any number of nearby UWB devices. The firmware is also able to multiplex a BLE connection with the phone while simultaneously performing low-power neighborhood discovery using a scalable rate-adaptive round-robin protocol for ranging.

We evaluated the performance of our system in many environments across four different buildings, including long corridors, different sizes of rooms separated by concrete, drywall, and various other construction materials. We tested in static as well as dynamic environments with moving people, furniture, and changing lighting. One of our tests includes five users moving around a large contiguous 3-floors area (30,000+ sq ft) within an office building. We moved furniture and toggled lighting in several tests to simulate more dynamic elements often found in the wild. In each test, the users walked freely, creating many non-line-of-sight (NLOS) scenarios with multiple walls between users. Across all of these experiments, Cappella provides a mean 3D geometric error performance of 0.9 m between users given different random walking paths. In addition, we observe that the quality of AR performance is sensitive to more than just geometric positioning error. Camera lens parameters, bearing, and distance combine to define visual registration errors that are highly dependant on the scene geometry. To better capture these effects, we also evaluate our system in terms of pixel error, which more accurately captures the visual displacement errors experienced by users. We observe that Cappella provides significantly lower pixel error compared to baseline methods. Our application source and UWB firmware are all open-source and available on GitHub.

Our core technical contributions are:

- An infrastructure-free multi-user AR system with real-time 6DOF positioning of users relative to each other.
- A distributed Rao-Blackwellized Particle Filter (RBPF) formulation and implementation that uses VIO and UWB readings complementarily to jointly estimate the user positions.
• An energy-efficient peer-to-peer UWB protocol with open-source firmware tailored toward wide-area relative positioning.
• An open-source end-to-end implementation and thorough evaluation of the proposed system. Our code is available on GitHub (https://github.com/WiseLabCMU/slam3d)

2 RELATED WORK

The topic of indoor positioning has received much attention over the past several decades for applications ranging from first responder tracking to autonomous drone navigation. In the context of multi-user AR, a key requirement is to be able to track several individuals relative to one another. There have been a plethora of approaches that satisfy this requirement, and we observe that they can be broadly classified based on how they establish a common coordinate frame between individuals: (1) *static infrastructure* systems, which use pre-placed, explicit infrastructure as a common reference, (2) *dynamic mapping* systems, which create maps of implicit infrastructure through environmental sensing and share these maps to provide a common reference, or (3) *infrastructure-free* systems, which forgo infrastructure entirely and instead use direct peer-to-peer measurements.

Systems based on static infrastructure have the fundamental limitation that they need to be installed *a priori*, so their use in AR is limited to environments that have been selectively prepared. Dynamic mapping systems relax this requirement by generating maps of the environment to use as impromptu infrastructure, but can be unreliable in dynamic environments where changes in lighting conditions or displacement of objects can make maps outdated or ambiguous. Infrastructure-free systems like Cappella enable AR applications in more general environments where mapping may be infeasible or ineffective.

2.1 Static Infrastructure Systems

There are many types of localization systems that rely on static infrastructure to establish a common coordinate frame. Outside-in tracking systems like OptiTrack use motion tracking cameras or other forms of active sensing in the environment to estimate the locations of several users. These systems are often expensive and are restricted to a small area of operation where the infrastructure is installed. Alternatively, visual fiducial markers, such as ARTags [30, 39] and AprilTags [29, 62, 67], are frequently used in AR systems to provide a reference between the physical environment and virtual objects. While these passive markers can be accurately localized with only a camera and low computational requirements, they only provide a location estimate when the tag is within the field of view of the camera, which means a dense deployment is required for wide-area coverage.

Beacon-based solutions provide continuous localization using UWB [44, 47, 57], BLE [15, 56], or ultrasound [24, 31, 33] ranging. These technologies are frequently combined with some form of odometry, either from an IMU or VIO, to smooth the location output and reduce the density of beacons required [18, 23, 35, 47, 52, 58, 61]. Cappella takes an approach similar to these systems, but rather than relying on fixed beacons in the environment, it instead uses peer-to-peer UWB ranges between users. This eliminates the need for any infrastructure, which enables AR applications to be untethered from specific physical spaces. Other works that take this approach of peer-to-peer ranging will be explored in Section 2.3.

2.2 Dynamic Mapping Systems

Simultaneous localization and mapping (SLAM) is a class of vision-based localization techniques for identifying and then leveraging features in an environment to track the position of a moving device. These methods use either monocular cameras [1, 5, 20, 40], depth cameras [4, 36, 66], or stereo cameras [9, 11, 65] to detect visual features from the scene, extract the 3D coordinates of the features, and determine the device’s 6DOF pose. These coordinates, however, are only relative to an arbitrary origin point that is not common across devices or across tracking sessions on the same device.

Collaborative AR and VR systems have been discussed in the academic literature as early as the late 90’s [8, 37, 46, 54], where specialized localization systems were used to combine coordinate frames. More recent developments in AR frameworks, such as Google’s ARCore, Cloud Anchors, Apple’s ARKit, and Microsoft’s Spatial Anchors have enabled multi-user capabilities. In these systems, each AR device individually performs SLAM to capture the visual features of the physical space relative to its local coordinate system. The users then share these visual maps to establish a common coordinate system and estimate the pose of other users. To share these maps between the users, Google ARCore uses a cloud-based architecture, which combines these maps centrally and sends the updated maps to all the users. Apple ARKit uses a peer-to-peer architecture, where the host of the AR session shares its current map with the users joining the session. However, any of these techniques impose significant communication overhead [53]. These maps consist of dense visual features, 3D meshes, or raw point clouds, which are usually large and difficult to transfer. In addition, the map matching algorithms assume a significant overlap between all of the users, which becomes unwieldy in terms of network traffic and computation in large areas. In many realistic scenarios, users often take disjoint paths through the environment, thus making map matching much more challenging and substantially increasing the convergence time.

2.3 Infrastructure-Free Systems

Traditional localization systems typically have the goal of estimating the "absolute" location in a fixed coordinate system that is mapped to the physical space using external systems. In this sense, the idea of "localization" is inherently tied to the existence of some form of infrastructure from which to base the measurements. However, reliance on infrastructure is infeasible in many AR scenarios, especially in the presence of multiple users. Instead, Cappella determines the relative positions between users to establish a common coordinate system for multi-user AR applications. For example, to display a virtual overlay of an object on the screen, it uses the knowledge of the object’s position relative to the user itself instead of anchoring it to the physical space.

The concept of relative localization has been used in sensor network localization for collectively locating stationary [41, 55] and mobile [21, 38, 51] nodes with respect to each other. These works provide the theoretical foundation for network localization using
We consider an indoor scenario consisting of \(N\) mobile users (nodes), all with unknown positions and orientations. The positions are unknown in 3D, but we assume that all nodes are able to measure the direction of gravity using their on-board accelerometers, which provides a common reference for two of the three orientation dimensions. Therefore, each node has four remaining degrees of uncertainty (three positional and one rotation about the gravity axis).

Each mobile user has an AR \textit{display device} and wants to position all other users, defined as \textit{target devices}, with respect to itself, without requiring any \textit{a priori} knowledge of the physical space or pre-installed infrastructure. The positioning framework has to work in real-time on limited compute platforms and needs to scale feasibly with the number of devices being tracked. All AR devices are equipped with two sensor systems:

- \textbf{VIO tracking:} Many devices that support AR, including smartphones and AR headsets, provide developer access to their internal VIO tracking. VIO tracks the motion of the camera by fusing detected visual feature points with inertial sensor data, including accelerometers and gyroscopes. The output of VIO is the position and orientation of the device with respect to the reference frame defined at startup, with the +\(y\) axis pointing towards the direction of gravity, which VIO is able to estimate using the accelerometer. Even though VIO provides the camera displacement over time, there is no common origin between multiple users to extract their relative positions, apart from a common gravity axis. Another challenge of VIO data is the accumulated drift error over time and sensitivity to environment conditions, such as lighting and motion.\[50\] contains a popular open-source VIO implementation; Cappella uses whatever source of VIO the AR platform provides.

- \textbf{UWB ranging:} Among various wireless ranging technologies that can penetrate obstacles (e.g., Bluetooth, UWB, WiFi, ultrasound), UWB is the most promising technology to combat multipath propagation in cluttered environments. As a result, we are seeing the appearance of UWB chips on the latest mobile phones, providing peer-to-peer ranging. However, each UWB-equipped device is only capable of measuring its distance to neighboring devices that are in range (<\(10–20\)m). Given the mobility of users, we cannot assume that range measurements occur synchronously or with any sort of regularity, resulting in sparse measurements that are difficult to use for real-time positioning. Additionally, UWB will occasionally provide erroneous measurements due to multipath in NLOS conditions.

\section{System Architecture}

Cappella adopts a scalable \textit{distributed architecture} in that each device computes the relative pose of its neighbors using peer-to-peer distance measurements. To deal with the sparsity of UWB readings, range measurements are combined with local camera VIO traces. The absolute nature of UWB ranging allows it to correct VIO drift over time. Finally, Cappella leverages the presence of multiple users and their mobility to collaboratively estimate the relative position of all users, improving the overall positioning accuracy.
while maintaining low computational overhead. An overview of
Cappella framework is depicted in Figure 2. Upon startup of an AR
app, the AR session tracks the pose of the device using VIO from
the AR API and begins collecting UWB ranges from neighboring
devices. These measurements are then passed to a particle filter
to estimate each device’s relative location. The following sections
elaborate on Cappella’s collaborative pose estimation technique
and the underlying challenges.

4 RELATIVE POSE ESTIMATION

Here we describe our relative positioning framework, which uses
a particle filter for tracking the N fark devices relative to the
display device. We start by explaining a simple approach that tracks
each target device independently and then demonstrate how it can
be enhanced by tracking all devices jointly, using Rao-Blackwellized
particle filtering (RBPF) to ensure that the problem remains tractable
as N grows.

4.1 AR Projection with Relative Coordinates

To project another user’s location onto the AR display, a reference
coordinate system is required. Since Cappella aims to do so in an
infrastructure-free fashion, we formulate the target device
relative to the current pose of the AR display rather than using
absolute coordinates. The pixel coordinates of the virtual object on
the display are defined as \([u,v]^T\):

\[
[u',v',w]^T = K^{-1}D_O^T V_O
\]

(1)

where \(D_O\) is a 4x4 matrix encoding the 6DOF pose of the display
relative to some arbitrary origin, \(V_O\) is a 4x1 vector encoding the
3DOF position of the target device relative to that same origin, and
K is a 3x4 matrix encoding the intrinsic properties of the virtual
camera such as resolution, and focal length \([32]\). We can simply
combine the first two matrices as:

\[
V_D = D_O^{-1} V_O,
\]

(3)

where \(V_D\) is now the 4x1 vector representing the position of the
target object relative to the display. As such, there is no require-
ment to explicitly track a global origin as long as all of the virtual
content is converted to the display device’s local coordinate system
before rendering. This can be achieved by simply tracking each
user’s position and orientation relative to the display device and
transforming that user’s local AR content accordingly.

4.2 Particle Filter (PF) Formulation

A particle filter for our state estimation has the following bene-
fits: (i) it is computationally easy to run online, (ii) it allows us to
use arbitrary noise models to describe VIO and UWB errors, (iii)
it can maintain multiple hypotheses when the solution space is
underdetermined, and (iv) it is agnostic to update rate, so range
measurements can be performed asynchronously while targets are
moving. It is notable that this approach allows a location estimate
to be available at the same rate as VIO updates, not just when
range measurements are performed. This means that AR content
can move accurately at a high framerate even when UWB ranges
are slow or temporarily unavailable.

4.2.1 State Space. We wish to track each device \(V_D^{(i)}\) relative to the
display \(D\). Each \(V_D\) consists of three positional components, \(x, y,\)
and \(z\). In addition, since the VIO estimates from each device are
with respect to a separate origin with a separate orientation, we
need to add components to the state space to track the orientation
of each device as well. By default, each VIO origin will have its \(+y\)
axis aligned with the gravity vector (up). Therefore, Cappella only
needs to estimate a single orientation angle \(\theta\) around the vertical
\((+y)\) axis for each \(V_D\).

Internally, VIO tracks the vertical direction using the device’s ac-
celerometer. While accelerometer bias, scale, and off-axis error can
cause slight pitch and roll deviation between devices, we found this
to not be a significant error in practice worth modeling in the estima-
tor. The yaw angle, however, is completely indeterminate between
devices with different starting orientations \(^2\). Thus, our state-space
for each tracked device has 4 dimensions: \(x^{(i)}, y^{(i)}, z^{(i)},\) and \(\theta^{(i)}\).

The particle filter works by sampling from this state space and
tracking the weighted samples as measurements become available.
Roughly speaking, VIO measurements for device \(i\) update the
positions of the samples in \([x^{(i)}, y^{(i)}, z^{(i)}, \theta^{(i)}]\) space and UWB
measurements between the display device and device \(i\) update the
relative weights of the particles according to their agreement with
the measured distance. These measurement functions are described
in more detail in the following sections.

4.2.2 VIO Measurements. VIO, like other forms of odometry, tracks
a device’s motion over time relative to some arbitrary origin. It mea-
sures \(dx, dy,\) and \(dz\). Although AR frameworks on mobile devices
normally perform loop closure to help mitigate drift, there is still a
steady accumulation of integration error that occurs in practice,
both in position and orientation about the vertical axis. Based on em-
pirical data collected from Apple ARKit and also the results shown
in \([52]\), we model these errors as Gaussian with small standard devia-
tions \(\sigma_{xyz}\) and \(\sigma_{\theta}\), respectively. This small amount of noise accounts
for minor errors in feature matching, scale estimation, and IMU
biases that are present in VIO measurements but are not modeled in
the particle filter. The state update equations for VIO at time \(t\) are:

\[
x^{(i)}(t+1) = x^{(i)}(t) + dx \cos \theta^{(i)} + dz \sin \theta^{(i)} + N(0, \sigma_{xyz}^2)
\]

(4)

\[
y^{(i)}(t+1) = y^{(i)}(t) + dy + N(0, \sigma_{xyz}^2)
\]

(5)

\[
z^{(i)}(t+1) = z^{(i)}(t) + dz \cos \theta^{(i)} - dx \sin \theta^{(i)} + N(0, \sigma_{xyz}^2)
\]

(6)

\[
\theta^{(i)}(t+1) = \theta^{(i)}(t) + N(0, \sigma_{\theta}^2)
\]

(7)

We note that, although the linear velocity error \(\sigma_{xyz}\) and rotational
velocity error \(\sigma_{\theta}\) are modeled as Gaussian in our formulation, there
may be some unexpected errors in VIO (such as large jumps) that
would fall in the tail of the distribution. We correct such errors us-
ing resampling techniques that account for the possibility of these
jumps (see “kidnapped robot problem” in \([59]\)).

4.2.3 UWB Measurements. UWB measurements occur frequently
but sporadically between pairs of nodes. They give a measurement
of the distance between a pair of nodes, with an error that is roughly
Gaussian with standard deviation \(\sigma_{z}\) \([52]\). However, consecutive
UWB measurements between the same pair of devices are not

\(^2\)While compass measurements can sometimes be used as an absolute yaw reference,
they tend to be unreliable indoors \([52]\).
perfectly independent, which breaks an assumption for Bayesian estimation techniques. This is due to systematic errors like antenna delay, clock frequency offsets between devices, and environmental conditions such as material penetration and multipath. Thus, using a Gaussian error model for these measurements in the particle filter can lead to false convergence and particle impoverishment, which are common issues for this type of estimator.

To combat these issues, we instead use a uniform probability model for UWB measurements. This way, several consecutive UWB measurements will not cause the particle weights to diminish as long as they fall within the bounds of the uniform model. The distribution extends \( \pm 3\sigma_r \) from the measured range, and we assume there is a \( P_{nlos} \) chance that the UWB range is entirely wrong due to NLOS errors. The probability model for obtaining a UWB range \( Z_i \) between the display device \( D \) and a target device \( V^D_{ij} \) is thus:

\[
P(Z_{ij}) = \begin{cases} P_{nlos} & \text{if } |\|\|_{D} - D \|_{D}^D - Z_i \| > 3\sigma_r \\ 1 - P_{nlos} & \text{otherwise} \end{cases}
\]

Figure 3: Cappella’s particle filter formulation jointly estimates multiple user positions (\( D \) and \( V_i \)) by combining UWB (\( Z_i \)) and VIO (\( U_i \)) measurements. The measurement dependency graph illustrates that each \( V_i \) is conditionally independent given \( D \), since each UWB measurement \( Z_i \) depends only on the pose of \( V_i \) and \( D \) (no UWB measurements are taken between \( V_i \) and \( V_j \) for \( i \neq j \)). Using Bayes’ rule, the joint distribution can be factorized as shown, resulting in the Rao-Blackwellized formulation in the box.

4.3 Collaborative Estimation with RBPF

A naive approach for tracking relative position of nodes is to define a completely independent particle filter for each AR node. As a result, the computational load scales linearly with the number of devices \( N \). However, since the particle filters are run independently, the model does not leverage the synergistic information that could otherwise be used in a collaborative formulation to mitigate the accumulated error due to noise in the display device’s own VIO tracking.

Alternatively, it would be possible to jointly model the states of all \( N \) moving devices. This way, every range could be used to improve the state estimation of all nodes in the joint distribution. However, sampling from \( 4N \) dimensional state-space would require a number of samples exponential in \( N \) in order to adequately sample the growing dimensionality, which would mean computation and memory requirements that scale exponentially with the number of users. A solution to this problem arises when some state variables \( Y^{(i)} \) are always conditionally independent given some other state variable \( X \). When this is the case, it is possible to factorize the joint probability distribution and independently track each \( Y^{(i)}|X \).

This approach, called Rao-Blackwellization (RBPF), is common in the SLAM literature [59] as a means of estimating a map whose elements are conditionally independent given a user’s location. As illustrated in Figure 3, our formulation of the relative positioning problem fits this framework, since UWB provides measurements of target device locations that are conditionally independent given the location of the display device.

In the RBPF formulation, a particle filter is used to represent the belief of the display device \( D \), where each target device \( V^D_{ij} \) can be represented by any probabilistic distribution. We chose to also represent the target device estimates using particle filters. Thus, the design of Cappella amounts to a “two-layer” particle filter. In the first layer, the state space of \( D \) is sampled to track the location of the display device. Then, for each of those samples, a second-layer filter is created to track each of the target devices given that sample. In each of these second-layer filters, the state space of \( V^D_{D} \) is sampled. In this way, the conditional relationship shown in Figure 3 is realized. In Section 6.7, we demonstrate the benefit of the collaborative nature of our joint RBPF formulation over the more common naive independent particle filter approach.

5 SYSTEM IMPLEMENTATION

There are three main components to the implementation of our system: a UWB ranging platform, an AR application with real-time positioning of all AR devices, and a large-scale ground truth collection system. The UWB ranging platform allows collection of range data between users in a dynamically sized ad hoc network. The AR application overlays digital objects on the estimated relative locations in the field of view of the display, allowing users to know where their teammates are without having a direct visual. It also collects ground-truth pose data by decoding AprilTags, which are placed strategically around the building to determine error in our system during data collection.
Figure 5: System components, including UWB ranging nodes, ground-truth markers, and mobile tablet application

5.1 UWB Ranging Prototype

Though not the focus of this paper, we realize that many localization researchers have struggled to find a UWB solution that can easily operate in peer-to-peer mode at scale. Unfortunately, most of the freely available reference implementations are designed for fixed infrastructure scenarios that may support mobile devices, as opposed to fully peer-to-peer operation. We imagine that phones with UWB hardware could eventually implement this functionality onboard once APIs become available.

A fully peer-to-peer ranging platform requires ad hoc ranging and neighborhood discovery. We developed an open-source and easy-to-use UWB firmware that provides these functionalities for the MDEK1001 modules from Decawave: (1) Neighborhood discovery using BLE’s GAP discovery protocol, (2) coordinated two-sided two-way ranging (DS-TWR) using UWB and (3) an interface to external systems using either USB serial or a standard BLE GATT server. We designed our protocol under the assumption that we have a highly dynamic mesh of nodes with hidden terminals and asymmetric links that change on the order of seconds. The MDEK1001 is an all-in-one battery- or USB-powered module with an enclosure that pairs a Nordic nRF52832 MCU with a DW1000 chip. The Nordic chip has a 64 MHz Arm Cortex-M4 processor with integrated BLE radio that can be programmed to act like a stand-alone node or pair with a mobile phone. Our firmware image exposes a standard serial interface (the AT command set) with the ability to store default parameters to flash memory, making it easy to configure addresses, sleep modes, neighborhood discovery polling rates, and UWB ranging options.

The neighborhood discovery protocol is BLE’s standard device discovery protocol. We allow users to define a custom advertisement period \( T_{BLE} \) (default period of 200 ms) and a configurable signal strength (RSSI) threshold for determining the most recent and closest neighbors. To save power when nodes are idle, we duty-cycle background scanning and disable the UWB radio. A node in the system can announce that it wants to participate in active ranging through its BLE advertisements. This in turn will wake-up nearby nodes and activate their UWB radios. Figure 4 shows an overview of the BLE and UWB transactions required to perform neighborhood discovery and ranging. Note that the BLE discovery modules use three channels and not just a single channel. Once activated, each node initiates a DS-TWR request (detailed in the upper right of the figure and in this application note [17]) over UWB at a user-configurable timing interval \( T_{UWB} \), with a default value of 100 ms. In each \( T_{UWB} \) period, the node performs a new DS-TWR request to the next node in its local neighbor list.

If DS-TWR messages are dropped, either due to collision or packet corruption, the next polling interval is randomly offset to avoid repeated collisions. We use an exponential random distribution across \( T_{UWB} \) similar in nature to slotted ALOHA [42]. As one would expect, as the number of neighbors increases, the polling rate of each individual neighbor decreases. We provide users with a lookup table for \( T_{UWB} \) values needed to support particular maximum node densities within a single collision domain. As shown in Figure 4, you can see that node \( B_1 \) transmits every \( T_{UWB} \) to node \( B_2 \), since it has no other neighbors. Node \( B_2 \) cycles through 3 total neighbors in its neighborhood list (the neighbor graph shown on the left). After nodes stop transmitting active ranging advertisements for a defined timeout, they return to their lower powered duty-cycled listening state. As shown in the bottom line of Figure 4, we also support simultaneously pairing an actively scanning node with a mobile device using a standard BLE GATT server. It is also possible to connect the MDEK1001 to a host device over USB serial or through its built-in RPI header. The default parameters of our firmware support 16-bit addressing (over 30K nodes) with cluster densities of 10 nodes at approximately 1 Hz update rates for each neighbor. Our low power sleep energy is on the order of 10 mW (mostly consumed by background BLE scanning) with an average active ranging energy of 800 mW. In practice, we see BLE neighbor discovery on the order of a 1-2 s with a typical 10-20 s eviction timeout. All source and documentation are available on GitHub (https://github.com/WiseLabCMU/Bluga/).

5.2 Prototype AR Application

We developed a prototype of Cappella as a mobile AR application running on iOS. This application provides two main features: (1) it shows the relative location and orientation of other users in the scene in AR (shown in Figure 5-c), and (2) it coordinates ground-truth data collection among mobile users (shown in Figure 5-b). The mobile app collects VIO data using Apple’s ARKit and UWB ranging data using a MDEK1001 module from Decawave over BLE. All ranging and communication information is shared using MQTT over WiFi, but this could conceptually be replaced by WiFi Direct or some similar peer-to-peer protocol. ARKit captures VIO data at 60 Hz and we collect UWB ranges with a polling rate of 10 Hz. As described earlier, the actual rate at which UWB data is received by each mobile user can vary and depends on the distance and number of neighbors around a particular node. Cappella is also resilient to message drops and reasonable levels of jitters (tens of ms). With message latency on the order of 100ms, it appears to perform well and is within common bounds for most single-hop wireless communication systems. It should be noted that in the current experimental platform, each node communicates using WiFi or LTE from the mobile device, but this could be easily replaced with WiFi Direct or other peer-to-peer protocols in a production implementation.

5.3 Ground Truth Collection

One of the biggest challenges for assessing the performance of a 6DOF positioning system at scale is accurately collecting ground-truth poses. We developed a data collection framework that periodically guides users to converge on "check-in" locations where...
AprilTags were used to accurately record 6DOF pose. We first installed over a dozen 8.5 by 11 in AprilTags across the multiple floors of our test buildings with retro-reflective markers on each corner. We surveyed the corners of each AprilTag using a total station with an advertised accuracy on the order of millimeters. In smaller experiments, we placed a number of AprilTags in fixed relative locations. To coordinate synchronized ground-truth readings between different users, we integrated an AprilTag decoder into the AR application, in which the users are instructed to move to the nearest AprilTag and wait until all users across the building had a high-confidence ground-truth measurement. Given the known tag location and the pose estimated by the AprilTag decoder, the application computes the ground-truth location, which is then published over MQTT to a central logging service.

6 EVALUATION

In this section we discuss our experimental setup, evaluation metrics, and perform a sensitivity analysis of a number of factors, including changes in lighting, background motion, user walking patterns, and RF non-line-of-sight conditions.

6.1 Experimental Setup

Our primary evaluation of Cappella consisted of a deployment across a 30,000 sq ft area spanning 3 floors of an office building, with 3 to 5 users walking in an arbitrary fashion, and 9 static nodes deployed for baseline comparison, as shown in Figure 6. We also stress tested Cappella in a diverse set of environments, both indoor and outdoor, different lighting conditions, as well as dynamic environments. The snapshots of these environments are shown in Figure 6. In all of these experiments, each user carried an iPad or iPhone with a built-in VIO tracking and a UWB node attached to the back of the device (as shown in Figure 5-a), while the static nodes consisted of just the UWB platform. As noted before, Cappella does not require any pre-installed infrastructure or static beacons for positioning, and here the static nodes are only used for our baseline comparison. Unless otherwise specified, all of our presented results only use ranges from mobile nodes.

The experiments consist of both LOS and heavy NLOS situations, with many instances where users are spread across 3 different floors with one or more dry/concrete walls between them. No instructions are provided to users on how to walk or how to hold the tablets. For 7 different experiments and 10-15 minute per run, the users walk with different speeds and periodically stand stationary, resulting in a total of about 40 minutes worth of data per person. This data is divided into an "evaluation" set, where users are walking normally, and a "sensitivity analysis" set, where users are walking in pre-defined patterns (evaluated in Section 7). As explained in Section 5.3, the ground truth was obtained with a number of AprilTags surveyed in a global coordinate frame using a total station. To synchronize the ground-truth measurements between users, the AR application guides users to scan a nearby AprilTag every 5-30 s over the course of each experiment.

6.2 Evaluation Metrics

It should be noted that the quality of AR performance is sensitive to more than just geometric error. Camera lens parameters, bearing, and distance combine to create the visual error seen by a user. To better capture these effects, we introduce an AR-specific metric, called display-proportional error (DPE), that combines distance, bearing, and the camera field-of-view as a single cohesive benchmark. We demonstrate the importance of this metric in AR applications in an example shown in Figure 7. 3 virtual cubes are overlaid at fixed distance from a set of (real) physical orange cones. The cones are located at distances of 1, 5, and 10 m, respectively, away from the camera. The green cube has no error, the yellow cube is offset by 0.5 m and the red cube is offset by 1 m. Notice that, due to perspective, the cubes that are further from the camera appear closer to the cone, even though their relative error in meters is the same. This simple example highlights why geometric error alone does not do justice to AR positioning performance. Instead, display-proportional error computes the AR error as the distance between an object’s true location and its estimated location when projected onto a 2D display, as a proportion of the display’s horizontal size. In the example in Figure 7, the closest yellow box has a DPE of 0.23 This error corresponds to approximately 1/4 of the screen width, while the farthest yellow box has a DPE of only .03, or about 1/33 of the screen width. In this sense, DPE captures the reprojected error of the estimated 3D locations, and can easily be used to calculate pixel error by simply multiplying by the display’s horizontal resolution. Therefore, we formalize our error metric definitions as:
3D geometric error: We calculate the average pair-wise Euclidean distance in 3D between all pairs of mobile nodes in meters.

Display-Proportional Error: Let \( \epsilon_{xy} \) be the \( xy \) component of the 3D geometric error, \( \epsilon_z \) as the \( z \) component, \( dist \) as the true distance between the display and the target object, \( f_x \) as the camera’s focal length (in pixels), and \( H_x \) as its horizontal resolution (in pixels):

\[
\epsilon_{xy} \left\| \frac{\epsilon_z}{dist + \epsilon_z} \right\| f_x \frac{1}{H_x},
\]
(9)

6.3 Baselines

We compare the performance of Cappella with two baselines:

1. **VIO-Only**: a typical infrastructure-free localization method [10] that uses VIO for pose estimation relative to the start point. This is a common localization method in robotics, but it requires initialization and \textit{a priori} knowledge of users’ start points. Even though this assumption is not feasible for most multi-user AR applications, it allows us to more easily isolate the performance contributions from VIO and UWB ranging in our system.

2. **UWB-VIO infrastructure-based oracle**: an infrastructure-based localization technique, which uses VIO to estimate 6DOF motion and UWB ranging to fixed beacons. We assume each fixed beacon (9 total) has a known global location in order to provide a baseline [61]. We consider this technique as our oracle and show that Cappella can achieve performance at nearly the same accuracy without relying on any of these pre-installed beacons.

6.4 Positioning Accuracy

We evaluate the positioning accuracy of Cappella across our evaluation dataset with 5 mobile users, on both single and multiple floors, and with a mixture of LOS and NLOS situations. Figure 8 shows the overall 3D relative localization error and compares it with our two baseline approaches. Cappella achieves a median 3D error of 0.9 m, compared to 2.5 m and 0.8 m in the VIO-Only baseline and UWB-VIO oracle, respectively. We can see that Cappella outperforms the VIO-Only baseline by leveraging the UWB ranging and collaborative pose estimation which mitigates drift over time. In addition, Cappella achieves relatively similar accuracy to the UWB-VIO oracle, which relies on pre-installed infrastructure and \textit{a priori} knowledge of beacons for trilateration that is unnecessary for Cappella.

As mentioned in Section 6.2, the 3D geometric error does not necessarily quantify the positioning performance relevant to AR applications. Figure 9 compares the AR performance of the three methods using DPE instead, which is a better representation of the pixel error the user will see in AR. In this context, a median DPE of 0.1 means that when the user is directly facing the physical target, the virtual target will be drawn only 10% of the display width away (128 pixels on a 1280x720 display, for example). By this metric, the virtual target will be at least somewhere on the screen whenever the DPE is less than 0.5.

6.5 Error vs. Separation Distance

Next, we evaluate Cappella’s performance as a function of distance. The ground-truth relative distance between users varies from 0.2 m to 27 m, including many instances of complete NLOS. Figure 10-a demonstrates the 3D relative error of each sample test (any pair of users at every 5 s interval) grouped by the ground-truth pair-wise distances. As we can see, error in positioning tends to increase slightly with distance, either due to UWB nodes going out of range or inherent VIO drift. However, unlike geometric error, DPE actually improves with distance. This suggests that a visual display showing an overlay with faraway users’ locations would still be effective at portraying those users’ locations.

6.6 Drift Over Time

In many positioning systems, including VIO tracking, error increases with time. Dead reckoning systems have inherent drift that is inevitable, and small errors in local motion estimation will eventually accumulate. As seen in Figure 12, Cappella is able to greatly mitigate drift and keep an almost constant error distribution over time using UWB measurements between devices, while the VIO-Only baseline exhibits a steady linear drift despite loop closure.

6.7 Impact of Collaborative Positioning

To evaluate the impact of our collaborative particle filter formulation, we compare the 3D relative error of the naive independent PF and collaborative RBPF, explained in Section 4. To isolate the impact of other parameters, including user mobility, number of users,
we perform controlled experiments with a single user and 9 static tags deployed for baseline comparisons. Then we estimate the relative location of static tags with respect to the user for different subsets of tags changing from 1 to 9 randomly selected tags. As seen in Figure 13, collaborative RBPF has a clear advantage over Independent PF. While with 1 static node, they are very similar in 3D relative error of around 0.9 m, the error of collaborative RBPF decreases from 0.85 m to 0.22 m with the addition of static tags.

This was expected as collaborative RBPF takes advantage of other system nodes’ estimates. Therefore, the drift of the mobile node is able to be somewhat mitigated by the averaging of noise across measurements to multiple other nodes. As more nodes are able to perform these “averaging corrections” together, the positioning system is able to converge to a more precise estimate than it could with nodes localizing individually. As a side conclusion, we can leverage this feature to further improve the positioning performance by deploying some static UWB tags with unknown locations. For example, in a first response operation, the users can deploy some static nodes at random locations as they move around the building to enhance their relative localization performance. Even though Cappella can operate completely infrastructure-free, it can nicely integrate with the infrastructure if one is present.

7 SENSITIVITY ANALYSIS

In this section, we elaborate on the computational overhead of Cappella’s collaborative localization algorithms. We also describe additional tests we performed in other campus environments and evaluate the sensitivity of Cappella to varying user mobility patterns and NLOS conditions in these environments.

7.1 Computational Overhead

Real-time computational cost is one of the critical factors of an AR positioning system, especially in mobile applications. Compared to independent particle filtering, our collaborative formulation achieves higher accuracy at the cost of higher computational overhead. Table 1, however, shows that Cappella can still operate in real time on a reasonable CPU. It should be noted that our implementation is not heavily optimized, and our compute time includes significant system overhead. The key takeaway is that the run-time overhead increases almost linearly with the number of users.

<table>
<thead>
<tr>
<th>Number of users</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Usage (%)</td>
<td>2.3%</td>
<td>8.0%</td>
<td>16%</td>
<td>25%</td>
</tr>
<tr>
<td>Memory Usage (MB)</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 1: Single threaded runtime performance on 2.4GHz i7 CPU
## 7.2 Performance Across Diverse Environments

In addition to the multi-story building tests described in Section 6, we also performed a series of tests across several other environments under different conditions, many of which are shown in Figure 6. These environments include: (1) a ‘busy’ office (with furniture being moved and lights being turned on and off to simulate ordinary office commotion), (2) a campus cafe with a large atrium and spiral staircase, (3) a hallway intersection near some elevators inside a brick building, (4) a dimly lit parking garage with height variation and lots of concrete and metal blocking line of sight, and (5) an outdoor area between campus buildings. The results of these experiments are shown in Figure 14. We see that performance is consistent across all of these environments, with the parking garage performance suffering slightly due to the heavy NLOS conditions and low light. Note that the performance in all of these environments is slightly better than the primary multi-story building test, which was the most challenging due to its immense scale.

## 7.3 Impact of Mobility Pattern

Another factor affecting the performance of Cappella’s positioning accuracy is the high dynamics of the environment and the mobility of users. To this end, we compared the system performance in 3 different walking scenarios: (1) when users were walking in pairs, which represents the near-best performance as the algorithm can take advantage of clean ranging estimates between each pair of users walking near each other, (2) normal walking when users randomly move in the space with a comfortable walking speed, and (3) when all of the users were performing fast movements, such as running, jumping, crawling, etc., for the purpose of stress testing the system. Figure 14 confirms the expected trend for different walking scenarios, and demonstrates that Cappella is resilient to fast motions and is therefore suitable for applications such as rescue operations or gaming.

## 7.4 NLOS Performance

Next, we study Cappella’s positioning performance in NLOS scenarios. Previous analysis shows that UWB ranging degrades in complete NLOS [52] due to noisy time-of-flight estimates that mainly capture multipath reflections instead of the direct distance between nodes. To evaluate this effect, we performed 3 different controlled experiments with different levels of NLOS. The first experiment includes 5 users that walk mostly in LOS of each other, all on the same floor. We then repeated this experiment with users walking in a larger space, including both LOS and NLOS conditions. Finally, we performed the experiment while users were spread out across 3 floors with some heavy NLOS conditions, such as multiple concrete walls between users, or being apart by more than 1 floor. As we can see in Figure 14, the 3D relative localization drops slightly with the increase of NLOS conditions, but we can still maintain a median accuracy of 1 m even in NLOS and extended ranges over 10-20 m.

## 8 DISCUSSION

In this section, we discuss the mechanisms to relax assumptions made in our current implementation of Cappella and the potential future extensions.

**Scalability:** While our current evaluations reached a maximum of 5 mobile users, Cappella’s collaborative approach should continue to improve in terms of localization accuracy (as shown in Figure 13) with even more users. This is mainly due to drift mitigation by averaging the noises across measurements to multiple nodes. However, eventually one would reach a computation and/or communication bottleneck. While our current implementation is not specifically optimized for scenarios with a large number of neighbors (many dozens), it would be possible to apply clustering heuristics based on user proximity. With a high density of users, it should be fairly easy to make sure that all clusters were at least partially connected. For applications where every user needs to know the location of every other user, each cluster would need to exchange their full state information with other clusters. We leave designing a highly scalable version of Cappella to future work.

**User Interactions:** In practice, Cappella works best when users occasionally pass near each other, resulting in high-confidence ranges and particle filter updates. So, the algorithm cannot benefit from collaboration if users are at the limits of the UWB range (100m in LOS and about 30m in severe NLOS). To avoid the performance degradation, one can add (arbitrarily placed) nodes. Such ‘bread-crumbs’ techniques have been widely proposed for rescue operations and are also compatible with Cappella.

**Darkness:** A limitation that is common among vision-based localization methods is sensitivity to low visibility conditions, such as smoke-filled rooms or extreme darkness. These conditions are commonplace for many search-and-rescue operations, such as firefighting. Our current experiments show that Cappella is resilient to partial darkness and dynamic conditions by leveraging the UWB ranging between users, but would still fail in total darkness.

Developing AR systems (even single-user) for these extreme conditions is challenging and an ongoing parallel research effort. Promising early results in 6DOF odometry systems that use infrared or millimeter wave sensing [19], which are inherently resilient to smoke, fog, and darkness, give us hope that an AR solution for emergency responders is on the horizon. When such an odometry source becomes available, we plan to integrate Cappella’s infrastructure-free relative localization framework to provide a robust multi-user AR solution.

**Gravity Estimates:** Cappella relies on VIO to provide orientation estimates that directly align with the gravity direction and provides no mechanism for automatically calibrating misaligned accelerometers. While small errors in VIO’s internal gravity estimation can be accounted for in the Gaussian noise model applied to VIO updates described in Section 4.2.2, there are certain situations where large errors may accumulate. For example, if users are riding in cars, trains, or elevators, the smooth acceleration may be misinterpreted as a change in gravity direction, which could cause integration errors in VIO. We have yet to characterize these errors, but they may become relevant in applications that involve navigation for transportation systems or in military use cases.
9 CONCLUSION

This paper proposes Cappella, a collaborative AR positioning system that allows multiple users to estimate their relative 6DOF poses in real-time. This system is free of infrastructure, is robust to environment dynamics and NLOS conditions, and maintains relatively low computational complexity to reduce power and update time. Cappella uses a form of Rao-Blackwellized particle filter to perform state estimation of the nodes jointly by using UWB ranging and VIO tracking. Cappella then displays the tracked nodes in an AR display in the coordinate frame of the user. Using the AR application, users can see where others are in the building despite walls, floors, and other obstacles creating NLOS conditions. We also present an AR metric that captures the quality of positioning with respect to the user’s display specifications, and is well suited for augmented reality applications.

As future work, we are interested in using the Cappella approach to bootstrap and correct mapped locations within fixed infrastructure systems. There is the potential to create a hybrid infrastructure-based and infrastructure-free AR positioning environment that could provide the best of both worlds where rapidly deployed relative content could persist in the environment once fixed infrastructure is encountered.

ACKNOWLEDGEMENT

This work was supported in part by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

REFERENCES


