TransTrack: Tracking Multiple Targets by Sensing Their Zone Transitions

Avinash Kalyanaraman, Erin Griffiths, Kamin Whitehouse
University of Virginia
{ak3ka, erg3wb, whitehouse}@virginia.edu

***Abstract*—In this paper, we consider a variant of the multi-target tracking problem in which the tracking region is divided into *zones* and targets can only be monitored as they transition between these zones. We call this the *transition tracking* problem. The key challenge in Transition Tracking is to estimate the number of targets in the tracking region without being able to sense all targets simultaneously. In this paper, we propose an approach to the Transition Tracking problem called *TransTrack*. Unlike most other tracking algorithms that maximize the likelihood of the sensor data, TransTrack applies penalty functions to find the minimum number of targets that can explain the sensor data. These penalties allow tracks with larger numbers of targets only if they have sufficiently fewer errors than other, alternative tracks. To evaluate this approach, we apply TransTrack to a dataset containing 3275 transitions between rooms in a home. We observe an average room tracking accuracy of up to 94.5%.**

I. Introduction

The multi-target tracking problem (MTT) [1] is essential to the functioning of many applications including air traffic control, robotics, and biomedical research. The most general form of this problem typically involves an unknown number of targets that move continuously throughout a region and that can appear or disappear [2]. Sensors estimate the positions of the targets at periodic intervals and also estimate identifying properties of the targets such as size, color, or shape. These measurements are subject to noise and the sensors may also generate false positives (a.k.a. false detections due to clutter) and false negatives (missed detections). In practice, the targets are typically observed with periodically scanning sensors such as a RADAR, an imager, or a LIDAR that can monitor the entire tracking region.

In this paper, we consider a variant of MTT in which the tracking region is divided into *zones* and targets can only be monitored as they transition between these discrete set of zones. We call this the *transition tracking* problem. This problem formulation is representative of an important set of real-world problems where complete coverage of the sensing region is not practical. For example, vehicle sensors are typically installed only at major intersections and do not cover the entire road network. Similarly, people sensors such as security cameras are typically installed at entryways and corridors but do not cover the entire building. As such, people and vehicles can be tracked as they transition between zones of the building or road network, but their position is not monitored while inside a zone.

Unlike traditional MTT, the sensors in transition tracking do not estimate the position of the target. Instead, they estimate *the destination zone* of the target as it passes through the transition area. Just like traditional MTT, sensors gather identifying properties of the target (e.g. size, color, or shape) for the purpose of data association, and are subject to three types of errors: sensor noise, false positives, and false negatives.

The key challenge in Transition Tracking is to estimate the number of targets in the tracking region without sensing all targets simultaneously, i.e. while sensing only the zone transitions of a target. The problem arises when spurious data produce a *phantom target*: a target that does not correspond to an actual object in the real world. Most tracking systems deal with phantom targets by periodically sensing all targets, e.g. with a radar scanner. In Transition Tracking, however, targets can remain (unobserved) in a zone for long periods of time and the tracking system cannot differentiate between a stationary target and a phantom target. Therefore, existing tracking algorithms that solve for a maximum likelihood solution will always overestimate the number of targets in the tracking region: phantom targets help explain spurious data, which increases the likelihood, but never generate data themselves and so never decrease the likelihood.

In this paper, we propose an approach to the Transition Tracking problem called *TransTrack* that jointly estimates the number of targets and their zone locations. Unlike most other tracking algorithms that maximize the likelihood of the sensor data, TransTrack applies penalty functions to find the minimum number of targets that can explain the sensor data. First, it creates a *target penalty* for having a larger number of targets in the tracking region, and applies this penalty only when a sensing error is observed. The intuition behind this approach is to allow tracks with larger numbers of targets only if they have sufficiently fewer errors than other tracks. Second, it creates a *mover penalty* for the number of targets that have moved since the last error. Again, this penalty is only applied when a sensing error is observed. The intuition behind this approach is to eliminate tracks in which different phantom targets are used to explain each sensor reading.

To evaluate, we modify a traditional multi-hypothesis tracking (MHT) algorithm to incorporate the TransTrack principles described above. We used the MHT to reduce computation time and TransTrack could also have been implemented as a Hidden Markov Model or Particle Filter. We then applied this implementation to a dataset created by the Doorjamb sensor [3], which is designed to sense the height and direction of people as they transition between rooms in a home. We use data from 3 controlled studies and 6 days of real-world in-situ deployment involving 2 to 3 participants and totalling 3275 doorway crossings. We observe an average room accuracy of 94.5% and 88.2% in the controlled and in-situ studies respectively.

II. Related Work

The problem of multi-target tracking (MTT) has been well explored by many prior works. We refer the reader to Blackman [4] and Pulford [5], for a survey of MTT methods. In this paper, we are interested in a variant of the MTT wherein only the transition of targets from one zone to another are sensed. One common approach is to directly apply a sequential Bayesian estimation algorithm such as the Kalman Filter, Hidden Markov Model (HMM), or Particle Filter [6], [7], [8]. However, these approaches choose a track by maximizing the likelihood of the data, which is not a viable approach for Transition Tracking because the number of targets is unconstrained and not all targets are sensed. Thus, the maximum likelihood solution will typically contain phantom targets in order to explain away any sensing errors.

Several other works have performed multi-target tracking by creating sensing zones. For example, Oh [6], Wilson [9], Kruger [10], Muller [11] treat the home as a graph of zones, each with its own sensors. However, these works assume that the sensors are located within the zones, whereas our work assumes that the sensors are located on the transitions between the zones. In other words, they assume complete sensing coverage of the tracking region. Additionally, those works assume the use of motion sensors, which do not have identifying information such as size, color, or shape. Thus, they do not need to address the data association problem in the same way. TransTrack deals with the additional challenges of observing target transitions rather than target states and, as a result, uncertainty grows quickly about both the number of targets and the state of each target.

The most similar solution to ours was developed for doorway tracking [3], [12], [13] - wherein the identity and direction of a target are sensed as it crosses the doorway. However, the solutions presented here assume a fixed number of targets with known identities, even though the occupants of a typical home come and go at different times, and occasionally bring guests into the home. Therefore, the algorithm that was analyzed did not need to address the complexity of estimating the number of targets in this environment, which is a key part to making the doorway tracking solution practical. Even if an application does not want to track guests, they can cause errors for resident tracking if the system cannot differentiate the guests from the residents. Consequently, there is a need for a tracking algorithm that tracks a variable number of targets by sensing only their transitions.

One common approach to the MTT problem is the Multi-Hypothesis Tracking algorithm (MHT), which is a deferred logic technique that delays uncertain data associations until more data become available by maintaining and scoring several alternative hypotheses. Originally developed for radar tracking systems [14] where the measured features are a set of discrete blips, it has since seen use in a diverse set of applications like pedestrian tracking [15], eddy current tracking [16], opponent player tracking in autonomous soccer robots [17] etc. However, inherent to these applications is the concept of periodically observing all targets in the tracking region, which is not a valid assumption in the transition tracking problem. To the best of our knowledge, this is the first work applying MHT in a multi-target transition sensing context.

III. Approach

There are two parts typically to a transition tracking system: (i) the *signal processing phase* which handles the raw transition sensor data and produces a discrete set of transition events (observations), and (ii) the *tracking phase* which operates on the output of signal processing and produces a discrete set of zone locations for each target. This paper focuses on the latter. The tracking algorithm must deal with any mistake made by the signal processing algorithm - viz false positives (FP), false negatives (FN), identity errors (IE) and direction errors (DE). Moreover, since we model a variable number of targets, the tracking algorithm must seek to prevent M targets from explaining away N-target data (where $M \neq N$). We point out that the tracking area could be frequented both by known identity targets (henceforth referred to as *dwellers*), and unknown identity targets (referred to as *visitors*). The tracking algorithm must be able to reason out between the two target types and track them.

We first define some terminologies. A *track* refers to a sequence of locations (zones) of each target. The *score* of a track denotes its quality. The *state* of a track contains enough information so that its *score* can be updated on each observation. It typically contains the current location of the targets. We next describe how the *state* of a track gets modified and its *score* evaluated in transition tracking.

Phantom Target Problem : In *Transition Tracking*, targets can remain stationary for long periods of time in a zone without being detected. Consequently, observations that are not easily explainable by existing targets will trigger the creation of phantom targets: spurious targets created by the tracking algorithm. For instance, consider a tracking area with two targets in it. As the two targets move, they cause sensing errors. We wish to prevent choosing those tracks which have extra (phantom) targets that sit idle and then move to explain these error events (caused by the two real targets). Such phantom targets can be brought into the tracking area by a track in several ways - e.g. (i) via a FP entry event, (ii) by treating the exit observation of a real target to be a FP and retaining the target, (iii) by treating the exit of a real target to be a DE, and bringing in another target, resulting in two phantom targets etc. As these phantom targets are unobservable until they actually move, there is no penalty in having them sit idle indefinitely, until an event occurs which no other target can explain. Indeed, the likelihood of any given data set will increase with a larger number of phantom targets. We refer to this as the *Phantom Target problem.*

To address this, we define an objective function that penalizes tracks based on the number of targets present, on non-compliance with an observation. The goal is to choose tracks with the minimum number of targets required to explain the observed data. We refer to this balance between phantom targets and unexplained observations as the *target-error tradeoff*.

Hidden Target Problem : However, such an objective function now suffers from the *Hidden Target* problem; since idle targets are unobserved, targets that are idle for a long period of time are evicted out of the tracking area with the goal to minimize the number of targets explaining the observations. To mitigate this, we incorporate a second penalty factor that penalizes a track based on the number of targets who actually move. In other words, a track does not get penalized for having idle targets. Using the two penalty factors together prevents tracks from having different phantom targets explain each observation. The notion of *mover penalty* can be incorporated in many ways: (i) by calculating the number of movers since the last non-compliance, (ii) by ranking targets based on their total number of moves, and then selecting the highest ranked mover since last non-compliance etc. We use the former. As a result, to capture this notion of movers, the *state* of a track is augmented with the *list of movers since last non-compliance*.

The non-compliance of a track with the sensor observation suffers a penalty depending on the error-type. We refer to such a non-compliance as an *'inferred error'* (e.g., if the observation says someone moved from zone z1 $\rightarrow$ z2, but the track moves a target from z2 $\rightarrow$ z1, then it has inferred a DE, and suffers a penalty). The other inferred error types are IE, FP and FN. Among these error types, IEs alone are not target-agnostic. Therefore, performing data association in the presence of *visitors* requires a notion of identity to be incorporated into a track's state. Addressing this by the inclusion of the identity vector for each target (containing its history of identity assignments) into track state achieves two goals: (i) data association for *visitors* can be performed by comparing the observed value with past values, (ii) prevention of a *visitor* from impersonating an *dweller* by better complying with the observations via techniques like T-Test. Summarizing, in transition tracking, for tracks to progress and be scored, the current location of targets, the list of movers since last non-compliance and identity vectors for each target become part of a track's state.

We capture these concepts via the approximate tracking technique of Multiple Hypothesis Tracking (MHT). Our implementation differs from classical MHT in that it performs multi-target tracking in the presence of infrequent observation of targets (transitions) as opposed to an entire field-of-view (FoV) scan.

IV. Implementation

In this section, we explain how we incorporate the TransTrack concepts into the classical Multiple Hypothesis Tracking (MHT) [14] algorithm. These concepts could equally be incorporated into other tracking algorithms, such as the HMM, but doing so greatly increases the state space. We chose to implement with the MHT because several heuristic algorithms enable computational tractability, albeit at the expense of optimality. To understand how the classical MHT must be modified to incorporate TransTrack concepts, we explain each of the key MHT steps below, including our modifications.

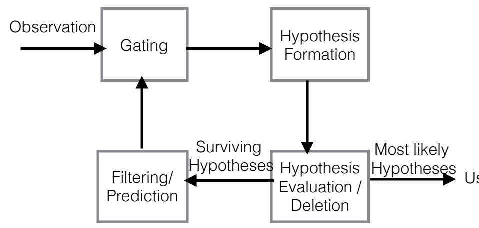

Fig. 1. The overall operation of a MHT (adopted from [4]).

Initialization: Let Z = $\{$*z0, z1, ... z(N-1)*$\}$ be the set of N possible zones a target can be in, with z0 denoting the *outside*. We abbreviate the tracking area (all zones besides the outside) as *TA*. Next, let T be the maximum number of targets trackable by the system. Let T_d be the number of *dweller* targets and T_g be the maximum number of visitors trackable, such that T_d + T_g = T. We define a *target state tuple* after the o^{th} observation to be a T-element tuple containing the zone location of the T-targets tracked - viz ρ_o = *(s1, s2, ... , sT)* where si $\in$ Z. Let δ_t be the identity vector of a target t - i.e. the list of identity values for the transitions assigned to target t. Let M be the list of movers since the last *inferred error*. A *hypothesis* H*i* refers to one possible explanation of the o observations, and thus exists as [(ρ_1, ρ_2, ... ρ_o) , (δ_1, δ_2 ... δ_T), M]. This can be understood as a sequence of target state tuples according to H*i*, identity assignments made to each target, and the list of movers since H*i*'s last inferred error. Thus, as per TransTrack's state definition, after the o^{th} observation, the o^{th} target-state tuple (most recent zone location of targets ρ_o), the identity vector for each target (δ_1, δ_2 ... δ_T) and the list of movers since last inferred error (M) constitute the *state* of a hypothesis.

On start-up, *TransTrack* starts with a blank slate, and considers each of the T targets to be equally likely in each of the N zones. This results in the creation of N^T initial hypotheses. The identity vectors of *dweller* targets are initialized to the known value, while those of *visitors* are set to ϕ. E.g., consider a two-target (T=2) case in a 3 zone state-space $\{$z0, z1, z2$\}$ with one *dweller* target (T_d=1) of known identity *id1*, and a maximum of one visitor (T_g=1). Then, two of the initial hypotheses are: H1 = *[[(z0, z1)], ([id1] ,ϕ), ϕ]* and H2 = *[[(z1, z2)], ([id1],ϕ), ϕ]*. H1 thinks only the visitor target is in the TA (in *z1*), and his identity is unknown yet. H2 is another hypothesis which thinks both targets are inside - one in *z1* and another in *z2*. Both hypotheses have no movers since last inferred error (since an error hasn't happened yet).

Figure 1 shows an overview of the classical MHT algorithm, adopted from [4]. We next describe how each block in this diagram behaves in *TransTrack*.

Gating: Gating determines if an observation can be physically caused by a target. In classical MHT, where the entire FoV is scanned, gating helps eliminate certain impossible data associations based on the kinematics of the moving object. However, in transition tracking gating is of little help because the sampling period (how often a target can be observed) is large relative to the potential speed of the target. E.g., a target can remain idle in a zone for 1 minute but it can also move to the other end of the TA via a small number of FNs within the same 1 minute. This makes most observations in the TA ambiguous. The presence of IEs exacerbates this problem. As a result, transition tracking does not benefit from gating as each observation become explainable by many targets.

Hypothesis formation: The hypothesis formation step is similar to conventional MHT. Here, the current set of hypotheses is extended by considering all possibilities. In transition tracking this means every observation causes each hypothesis to duplicate itself upto (2T+1) times and progress as:

1) Someone who is inside and has the transition area within his gate, has moved through it in either direction
2) Someone who is outside and has the transition area within his gate, has come in, and moved through it in either direction
3) Observation was a false detection

For example, consider a 3-target scenario in a 3-zone state space (z0 $\leftrightarrow$ z1 $\leftrightarrow$ z2), where z0 $\leftrightarrow$ z1 is the exterior transition sensor. Upon detecting a z1 $\rightarrow$ z2 transition event with observed identity id_{obs}, a hypothesis ending in target-state tuple (z1, z2, z0), say [... (z1, z2, z0) , (δ_1, δ_2 , ϕ), {t1}] would duplicate itself (2T+1) times and progress them in the following way:

$$H1 : [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \{t1\}] \xrightarrow{FP} [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \phi]$$
$$H2 : [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \{t1\}] \longrightarrow [...(z2, z2, z0), (\delta_1', \delta_2, \phi), t1]$$
$$H3 : [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \{t1\}] \xrightarrow{1FN, DE} [...(z1, z2, z0), (\delta_1', \delta_2, \phi), \phi]$$
$$H4 : [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \{t1\}] \xrightarrow{1FN} [...(z1, z2, z0), (\delta_1, \delta_2', \phi), \phi]$$
$$H5 : [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \{t1\}] \xrightarrow{DE} [...(z1, z1, z0), (\delta_1, \delta_2', \phi), \phi]$$
$$H6 : [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \{t1\}] \xrightarrow{1FN} [...(z1, z2, z2), (\delta_1, \delta_2, [id_{obs}]), \phi]$$
$$H7 : [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \{t1\}] \xrightarrow{2FN} [...(z1, z2, z1), (\delta_1, \delta_2, [id_{obs}]), \phi]$$
$$where, \delta_1' = \delta_1 \oplus id_{obs} \ , \ \delta_2' = \delta_2 \oplus id_{obs}, \ and \ \oplus \ denotes \ append$$

H1 is the hypothesis that thinks the observation is a false detection. H2, H3, H4 and H5 move the two targets inside the TA through the (z1,z2) sensor in either direction. H6 and H7 hypothesize that some target from the outside has come in and transitioned in either direction. Hypotheses H3, H4, H6 and H7 think that some observations have been missed (FN). Note that each hypothesis also appends the observed value to the identity vector associated with the hypothesized mover. We also point out that the list of movers since last inferred error gets reset ({t1} becomes ϕ) whenever an error is inferred.

Next, each hypothesis explores the possibility that someone inside has exited the TA after the current observation via a missed detection. For example, H7 duplicates itself three times, and advances them the following way:

$$H8 : [...(z1, z2, z1), (\delta_1, \delta_2, [id_{obs}]), \phi] \xrightarrow{FN} [...(z0, z2, z1), (\delta_1, \delta_2, [id_{obs}]), \phi]$$
$$H9 : [...(z1, z2, z1), (\delta_1, \delta_2, [id_{obs}]), \phi] \xrightarrow{2FN} [...(z1, z0, z1), (\delta_1, \delta_2, [id_{obs}]), \phi]$$
$$H10 : [...(z1, z2, z1), (\delta_1, \delta_2, [id_{obs}]), \phi] \xrightarrow{FN} [...(z1, z2, z0), (\delta_1, \delta_2, \phi), \phi]$$

Note that the *visitor* target's identity vector is reset on exit (H10). This is to capture the intuition that no two *visitor* targets are necessarily the same. Given that there are N^T initial hypotheses, $(N^T) * (T * (2T + 1))^D$ hypotheses are formed after *D* events. This exponential explosion of hypotheses necessitates track pruning.

Hypothesis evaluation/deletion: Temporal pruning techniques such as *n-scanback* [16], [18], [17] which are commonly employed in conventional MHT, cannot be applied in our case as a target can remain idle in a zone for an indefinite amount of time. The intuition behind *n-scanback* is that ambiguities get resolved in atmost *n* scans. Secondly, given the large gating challenge, *n* cannot be large, as it will result in storing a large number of hypotheses. For a TA of 8 zones and at most 4 targets, *3-scanback* itself results in the maintenance of over 50 million hypotheses.

Therefore, we develop an alternate two-step pruning strategy that leverages the discretization of states. First, if two hypotheses have the same current zone location of each target, same movers since last inferred error and same identity statistic (e.g. mean of identity vector) for each target, then only the better one is retained (i.e. either H1 or H10 in the above example). We refer to this as *equal state* pruning. To further keep the state space tractable and maintain enough diversity and coverage across all possible zone locations, we maintain the top-M (M=4) hypotheses ending in each possible target state tuple. This results in the constant maintenance of $M * N^T$ hypotheses. However, choosing one hypothesis over another necessitates a scoring function.

Score function: Scoring in MHT is done in an application-specific manner depending on the constraints of the problem. The aim of our scoring algorithm is to address the *phantom target* and *hidden target* problems. As a result, each hypothesis on non-compliance with an observation suffers a penalty depending on the number of targets present and the number of movers since the last inferred error. More formally, a hypothesis after the i^{th} observation gets penalized according to the following score function:

$$pen(i) = pen(i-1) + \alpha * \sum\nolimits_{j \in E} e_j w_j \tag{1}$$

where: E : set of inferred errors {FP, FN, DE, IE}
e_j : error penalty associated with the inferred error
w_j: weight of the inferred error type
α : correction term for target-error tradeoff = (m + t + k)
t : number of targets in the TA during the error
m : number of movers since last inferred error ($m \leq t$)
k : constant offset to eliminate bias towards certain tracks

We next explain each term in the above score function:

e_j *Error Penalty* – Each noncompliance by a hypothesis with an observation suffers a penalty depending on the inferred error type. *TransTrack* makes use of a probability value passed up from signal processing, whenever available, and a unit penalty otherwise. FNs suffer a penalty equal to the minimum distance to the transition sensor. We infer IEs for *visitor* targets by comparing the observed value with the visitor's maintained identity value (e.g. mean), based on any past observations. Since, a visitor's identity is learnt on-line, it can *impersonate* an *dweller* by better complying with the observations, resulting in incorrect data association. To eliminate this, each hypothesis upon exit of a visitor performs an equal variance T-test (p = 0.05) between that visitor's identity vector and the identity vector of each *dweller* who has been outside since the visitor's

Time	$\tau0$		$\tau1$	$\tau2$	$\tau3$	$\tau4$	$\tau5$	$\tau6$	$\tau7$	$\tau8$		$\tau9$
GT track (H1)	(z1, z2)		(z1, z3)	(z1, z4)	(z1, z3)	(z1, z3)	(z1, z4)	(z1, z5)	(z1, z4)	(z1, z3)		(z2, z3)
Inferred error of H1	-		DE	IE	-	FP	-	DE & IE	-	IE		-
Score (H1)	0		3	6	6	9	9	15	15	18		18
Chosen Track (H2)	(z1, z2)	(z0, z2)	(z0, z3)	(z0, z4)	(z0, z3)	(z0, z3)	(z0, z4)	(z0, z5)	(z0, z4)	(z0, z3)	(z1, z3)	(z2, z3)
Inferred error of H2	-	FN	DE	IE	-	FP	-	DE & IE	-	IE	FN	-
Score (H2)	0	3	5	7	7	9	9	13	13	15	17	17

TABLE I

EXAMPLE OF *Hidden Target Problem*: GT TRACK (H1) GETS EVICTED BY AN ALTERNATE HYPOTHESIS (H2) AT TIME $\tau9$. H2 HAS A LOWER SCORE BECAUSE THE IDLE TARGET WAS EVICTED AFTER $\tau0$, AND BROUGHT BACK IN BEFORE $\tau9$.

entry. A p-value greater than 0.05 implies that the identity distributions are similar, and the visitor hypothesis gets evicted.

w_j: *The weight of the inferred error type* – This term captures the likelihood of each error type across different error types. This is a sensor property. E.g., if missed detections are less likely than false detections, then $w_{fp} < w_{fn}$.

t: *The number of targets in the sensing-area during the error* – This term is used to address the *Phantom Target* problem. To eliminate any bias on the hypothesis which has all targets outside (i.e. t = 0), we use $(t{+}1){*}\Sigma e_j w_j$. As mentioned earlier, using the target-factor alone in scoring results in hypotheses being subject to the *'hidden target problem'*. Table I shows an example of this problem with two known-identity targets *t1* and *t2*. Let us say that the arrangement of the zones are (z0 $\leftrightarrow$ z1 $\leftrightarrow$ z2 $\leftrightarrow$ z3 $\leftrightarrow$ z4 $\leftrightarrow$ z5), where z0 denotes the outside-zone. $\tau0$ is the time that target *t1* becomes idle at zone z1 and $\tau9$ is the time *t1* moves out of z1. At $\tau9$, hypothesis H2 evicts H1 with a lower score. This is because H2 having incorrectly evicted the *hidden* target *t1* out of the TA, ends up with a lower total penalty. For brevity sake the identity vectors are not shown, as they are always identical for H1 and H2.

We next calculate the bounds for the eviction of a hidden target - i.e. the minimum number of errors an idle target can tolerate before being evicted by a hypothesis with a lower score. For simplicity of derivation, we assume uniform unit weighting. Let τ_a be the time that *t1* becomes 'idle', and τ_b be the time that *t1* makes a transition again. Let *d* denote the minimum number of hops from the 'hidden' zone to the *outside*. Between τ_a and τ_b, targets t2 to tT move and cause *e* errors. We wish to compare two hypotheses, H1 : *t1* rightly remains idle, and H2 : *t1* gets evicted via FNs just after τ_a, and brought in via FNs just before τ_b.

$$\begin{aligned}
\text{Score (H1)} &= \ e \text{ errors caused with T targets in TA} \\
\text{Score (H1)} &= \ (T+1) \ * \ e \\
\text{Score (H2)} &= \ FN \text{ to evict } t1 \\
&\quad + e \text{ errors caused with (T - 1) targets in TA} \\
&\quad + \ FN \text{ to enter } t1 \\
\text{Score (H2)} &= (T+1)*(d) + ((T-1)+1)*e \\
&\quad + (T+1)*(d)
\end{aligned}$$

To retain *t1*, score(H1) $<$ score(H2). Therefore,

$$\begin{aligned}
&(Te+e) < 2d*(T+1) + Te \\
&\quad \implies e < 2d*(T+1)
\end{aligned} \tag{2}$$

Consequently, in a 2-target case, a static target in the leaf-node (d=1) gets evicted after 6 errors of the other target.

m: *The number of movers since last inferred error* – This term is used to mitigate the *Hidden Target* problem. As before, to eliminate any bias towards an all FP track (i.e. m = 0), we use the factor of $(m+1)$. This results in our score formula of: $(t+m+k)*\Sigma e_j w_j$ where k = 2.

Study	Participant Heights (m)	# crossings
Controlled Study1	1.63 , 1.80	400
Controlled Study2	1.63 , 1.80	398
Controlled Study3	1.52 , 1.63 , 1.75	516
In-situ study (6days)	1.52 , 1.88	1961

TABLE II

EXPERIMENT DETAILS: A TOTAL OF 9 STUDIES WITH 3275 DOORWAY CROSSINGS INVOLVING 2 TO 3 PARTICIPANTS WAS PERFORMED

We next analyze this scoring function using the same notation. For the sake of simplicity in the derivation, let us consider that every target except *t1* moved between each of the *e* inferred errors. This consideration is just for ease of understanding as the derived inequality is independent of the actual number.

$$\begin{aligned}
\text{Score (H1)} &= \ e \text{ errors caused with T targets in TA} \\
\text{Score (H1)} &= \ (T+T+2) \ + \ ((T-1)+T+2) \ * \ (e-1)) \\
\text{Score (H2)} &= \ FN \text{ to evict } t1 \ + \\
&\quad e \text{ errors caused with } (T-1) \text{ targets in TA} + \\
&\quad + \text{FN to enter } t1 \\
\text{Score (H2)} &= (1+T+2)*(d) + ((T-1)+(T-1)+2)*e) \\
&\quad + (1+T+2)*(d)
\end{aligned}$$

To retain *t1*, score(H1) $<$ score(H2). Therefore,

$$\begin{aligned}
&(2+2T) + (2T+1)*(e-1) < (3+T)*(2d) + 2Te \\
&\quad \implies e < ((3+T)*2d) - 1.
\end{aligned} \tag{3}$$

Comparing inequalities 2 and 3, it can be shown that *((3+T) * 2d) - 1* $>$ *2d*(T+1)*, since d >= 1, confirming that the bounds have increased. In a 2-target case, a static target in the leaf-node now gets evicted after 11 errors of the other target.

In order to avoid growing memory costs, we define a commit policy. After every observation, if all hypotheses agree on a common prefix (i.e. they agree on the zone-locations of each target, from event E_0 to E_i), then the prefix is committed to disk. Subsequent prefix checks happen from event E_{i+1}.

At any time instant, the lowest scored hypothesis is the best hypothesis. We point out that the recursive nature of the score function makes it unnecessary to have the complete dataset to generate zone estimates, making *TransTrack* conducive for near real-time tracking.

Filtering/Prediction: In classical MHT, every hypothesis uses a motion model to predict the location of each target for the next scan. However, in transition tracking because of (i) the possibility that a target can stay in a zone for an indefinite amount of time, and (ii) the inevitability of large gates, no prediction is made on the next location of a target.

V. EXPERIMENTAL SETUP

We evaluate our tracking algorithm with a doorway tracking application using a Doorjamb-like sensor setup [3] in a detached home of 9 rooms (Figure 2) involving 2 to 3 persons (targets). The system is mounted on top of each doorway and measured the height and direction of a target as they transitioned through the doorway. We perform 3 controlled studies and 6 days of real-world in-situ deployment. The *diameter* of the house was 4. This meant one could move from one end of the state space to the other with just 4 FNs, making each doorway transition event became explainable by any of the targets. Table II describes details of each study and its participants. The first two controlled studies had the same participants. They were asked to leave all doors open in Study1, but open and close doors as they performed the experiment in Study2. This was to study the effect of errors on tracking, as the movement of doors lead to signal processing errors. Controlled Study3 had no constraints, and the participants were asked to enter, exit and walk around as naturally as possible. Ground truth for the study was collected using cameras installed on the doorway. To ensure that participants lived naturally, the field of view of the cameras were restricted in hardware to only the doorjamb of the doorway. The recorded video was processed, to extract the identity and direction of participant involved in the crossing.

We evaluate tracking using the *Room Accuracy* metric. This metric evaluates if a person is ever detected in the correct room during the time he was in that room. This is calculated as the *F-score* of the *room recall* and *room precision*. *Room recall* is defined as the fraction of the total number of room occupancy periods (the time a person is in a specific room) in ground truth where tracking also correctly placed the same person in the same room at least once during that occupancy period. *Room precision*, the complement to *room recall*, is defined as the fraction of the number of room occupancy periods in tracking, where ground truth also had the same person in the same room at least once during that period.

We compare *TransTrack* against three baselines that can track a variable number of targets and have the same set of requirements as *TransTrack*. Our first baseline *Nearest Identity* is a stateless approach to tracking that moves occupants based on the identity data observed at the doorway, with no regard to his previous location. It chooses an occupant based on the height measurement and puts him into a room based on the observed direction. Our second baseline, *Nearest Neighbor* is a well-known stateful greedy target tracking approach [19], [4]. Each observation is assigned to the occupant closest to the doorway with heights used as a tie-breaker. The location of the occupant is updated after an assigned observation. Our third baseline *K-best* is a variant of [20] and maintains the K lowest scored hypotheses after every observation. The K-value was chosen such that its time and space complexity were identical to that of *TransTrack*. No equal-state pruning is performed here, but the score function is identical to *TransTrack*. To have all algorithms on an equal footing, they are all started with a known initial state of the home.

Finally, we use uniform unit weighting on all errors except FNs. FNs have twice the weight for two key reasons : (i) the signal-processing recall of our system is better than signal-processing precision, and (ii) to increase the bound on the *hidden target problem*. We start off by tracking a maximum of 4 targets in a home (i.e. T = 4), and then study the behavior as we vary the maximum number of targets tracked. We refer to the targets with known identities as *residents*, and those with unknown identity as *guests*.

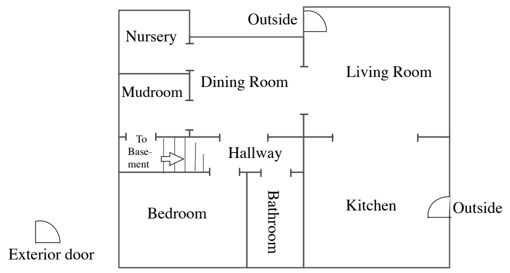

Fig. 2. Experiment scenario: Floorplan of experiment home with 9 rooms and 2 exterior doors used to evaluate TransTrack

VI. Results

Figure 3 shows that *TransTrack* achieves the highest average accuracy of 94.5% and 88.2% in the controlled and in-situ studies respectively. The *Nearest Identity* approach's average accuracy of 86.2% and 78.3% in controlled and in-situ respectively is lower than *TransTrack* because it does not use the future to disambiguate the past. The lower 73.2% and 65.1% average accuracy of the *Nearest Neighbor* approach can also be attributed to the lack of use of future. However, the maintenance of state exacerbates the problem here. *K-best* has an average accuracy of 88.2% and 64.5%. It is lower than *TransTrack* because of the absence of equal-state pruning. Consequently, the K-best hypotheses have many hypotheses with an equal last state, evicting out the desired hypotheses. An average of 394397 hypotheses were pruned after every observation via the equal state pruning strategy.

We also performed two additional analyses: (i) On comparing *TransTrack* with *Doorjamb's* algorithm, we noted that it performs almost as well as *Doorjamb* (average less than 5% off) even without assuming the number of targets at all times. (ii) We also evaluated the algorithms by calculating the average resident room accuracy, after making each of the residents as guests. We noticed an identical trend to Figure 3 with *TransTrack* suffering an average accuracy drop of 3.8%.

Figure 4 shows how different parts of the scoring function affect in-situ tracking accuracy. Simply maximizing the likelihood of the observations gives only 77.5% accuracy. This is because of the presence of phantom persons who sit idle and move to explain the errors of the real targets. The addition of a mover penalty alone does not increase the accuracy as it still suffers from the phantom effect. However, the addition of a target penalty alone increases the accuracy to 81.6%. This increase is because there is now a penalty of having extra idle persons in the home. However, such an approach suffers from the *hidden target problem*. Adding the mover and target penalty together increases the accuracy to 83.5%. As previously stated, a guest can still impersonate a resident by complying with the observed heights better. We see that

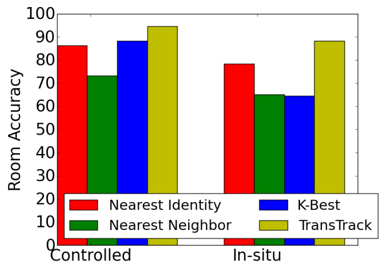

Fig. 3. *TransTrack* consistently performs better than the baselines. *Nearest Identity* and *Nearest Neighbor* approaches suffer as they do not use future information. The absence of equal state pruning affects accuracy of *K-best*

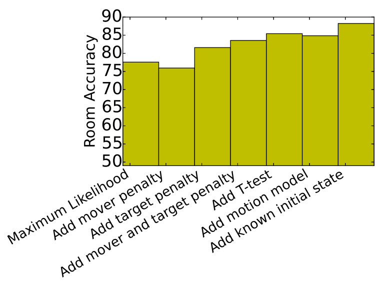

Fig. 4. Scoring function variants : The addition of target penalty, mover and target penalty, explicit guest reasoning and knowledge of initial state all increase in-situ room accuracy over the maximum likelihood approach

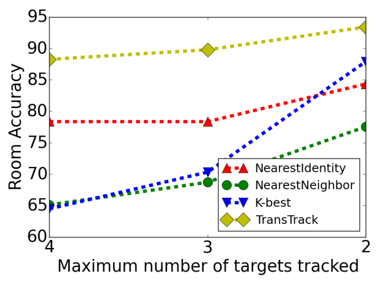

Fig. 5. Sensitivity analysis on the maximum number of targets (T) tracked: As T decreases, the accuracy of all algorithms increase. However, TransTrack still continues to perform better than other baselines.

the subsequent addition of explicit guest-resident reasoning via the T-Test increases the accuracy to 85.4%. We next notice that the subsequent addition of a motion model does not increase accuracy. The motion model is that it takes a target at least 1 second to pass through every room. In other words, no hypothesis exists wherein a target *t* can explain the doorway event of another target *t'* which is H hops away, if the last moved time of *t* is less than H seconds. Such a motion model does not increase accuracy because we noticed that the percentage of concurrent moves by persons is low, and even during the times of concurrent moves *TransTrack* was already doing a reasonable job in associating the observations to targets. Finally, we notice that starting at a known initial room location for each person increases the accuracy to 88.2%.

We next calculate *room accuracy* as we start varying the maximum number of tracked targets (T) from 4 to 2. As seen in Figure 5, decreasing T increases the accuracy of all algorithms. This is because each observation can potentially be explained by a lesser number of persons resulting in lesser ambiguity. *TransTrack* which achieves 93.4% and 89.8% with T = 2 and T = 3 respectively, still continues to consistently perform better than the other baselines. Since the same trend can be seen in the controlled study too, in the interest of space, we show the graph for in-situ alone.

One of the main reasons for the performance difference between in-situ and controlled studies was due to error-clustering (bursts of errors). The boxplots in Figure 7 show the number of FPs or FNs in any 20 event window for each data set. It is seen that in in-situ, there exists several cases in which more than half of the events in a 20-event window are either FPs or FNs. Since *TransTrack*'s scoring assumes a uniform error distribution, when such bursts of errors happen, it tries to explain these erroneous events with extra persons, as the height and direction estimates are inconsistent with the targets at home. Consequently, the accuracy suffers. These error bursts were mostly due to a target moving back and forth near a doorway. Other causes were due to crouching, moving hurriedly etc.

Next, we study the effect of removal of each error type on tracking accuracy. Figure 8 shows that an increase in accuracy is generally observed with the removal of each error type, approaching 100%. This is because as errors get removed, there is lesser ambiguity in data association for *TransTrack*.

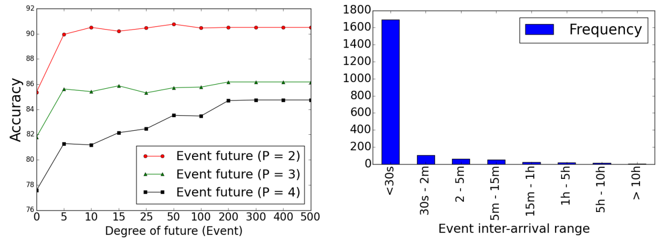

Fig. 6. Degree of future analysis: Most of the ambiguities get resolved within 5 future events. As the inter-arrival time between nearly 90% of events is less than 30 seconds, this roughly translates to within 150 seconds of future.

Since, *TransTrack* uses the future to disambiguate the past, we calculate how much future is required to correctly resolve a doorway crossing event, in the in-situ study. Figure 6 shows that most ambiguities can be resolved within 5 future doorway crossing events. To quantify this in terms of time, we looked at the inter-arrival time between consecutive doorway events. We noticed that nearly 90% of events arrive within 30seconds of the previous event. This effectively means 5 future events roughly translates to around 150 seconds of future. These results indicate that *TransTrack* could support applications such as HVAC control.

We next calculate another metric, *Target Count Accuracy*, defined as the fraction of observations after which tracking and ground truth had the same number of targets. To match the ground truth doorway events with the tracking events, we use the data-fusion algorithm of Kalyanaraman [21]. Figure 9 shows that *TransTrack* achieves nearly 30% better accuracy than any baseline. The Maximum Likelihood approach gave nearly 0% accuracy. TransTrack performs better than the rest as it addresses the *Phantom* and *Hidden Target* problems.

#GT Targets	#TransTrack Targets				
	0	1	2	3	4
0	**7**	25	9	5	2
1	6	**154**	47	11	0
2	1	12	**1210**	379	99

TABLE III
CONFUSION MATRIX COMPARING THE NUMBER OF TARGETS IN TRANSTRACK WITH GROUND TRUTH(GT), AFTER EACH TRANSITION. TRANSTRACK RARELY UNDER-ESTIMATES AND OVER-ESTIMATIONS ARE DUE TO ERROR CLUSTERING (EXPLAINED IN FIGURE 7)

Table III shows the in-situ confusion matrix comparing the

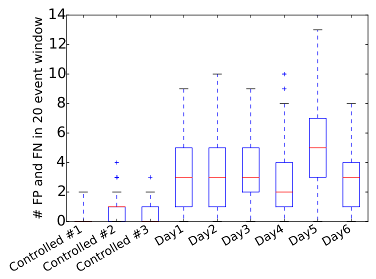

Fig. 7. Even though the average precision and recall for in-situ was comparable to the controlled studies, the in-situ data has many bursts of 8 or more false positives or negatives within a 20-event window. These bursts help explain the difference in tracking accuracy between the two studies.

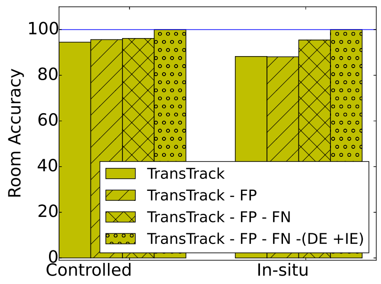

Fig. 8. Effect of error removal: As errors get removed, there is generally an increase in accuracy owing to lesser ambiguity

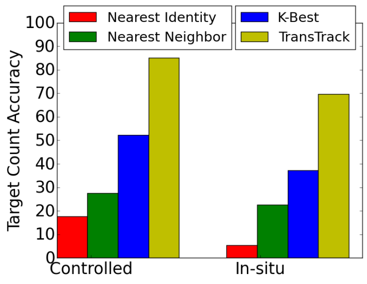

Fig. 9. TransTrack which addresses the Phantom Target and Hidden Target problems tracked the correct number of targets in a home better than any of the baselines.

number of targets in the home in ground truth with *TransTrack* (in terms of number of transitions). It is seen that *TransTrack* rarely under-estimates. The over-estimation can primarily be attributed to the error-clustering of Figure 7 (explained earlier). However, these extra targets mostly remain idle, and eventually end up getting evicted out of the house.

We point out that there is an inherent accuracy-complexity trade-off in using *TransTrack*. Even though *TransTrack* achieves higher accuracy than the greedy baselines, it trades-off higher time and space complexity as it retains state information to help in disambiguation. However, this is typically not an issue for transition tracking in domains like homes, given the low number of targets at any given time.

VII. Conclusion

In this paper, we present TransTrack, an approach to track a variable number of targets by sensing only their transitions. The presence of sensing errors and large sampling period relative to the potential speed of the target leads to uncertainty in the number of targets. We show the existence of a fundamental tradeoff between the number of targets tracked and the sensing errors they cause. Our evaluation of *TransTrack* on 3 controlled studies and 6 days of real-world in-situ data showed that it consistently performed better than the baselines.

We believe that the findings presented here will become more important with time as more diverse and non-invasive sensors get deployed. For instance, we envision the evaluated doorway tracking system to be augmented with motion sensors, which can observe state. This warrants the need for novel fusion tracking algorithms with transition sensors observing identity, and state-observing sensors detecting presence. Such algorithms could augment our findings with the well-studied state sensing literature [6], [4], [18].

VIII. Acknowledgements

We thank the anonymous reviewers, Nipun Batra and Vijay Srinivasan for their valuable feedback.

References

[1] M. Mallick, B. Vo, T. Kirubarajan, and S. Arulampalam, "Introduction to the issue on multitarget tracking," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 7, no. 3, pp. 373–375, 2013.

[2] S. Oh, S. Russell, and S. Sastry, "Markov chain monte carlo data association for general multiple-target tracking problems," in *CDC*, '04.

[3] T. W. Hnat, E. Griffiths, R. Dawson, and K. Whitehouse, "Doorjamb: unobtrusive room-level tracking of people in homes using doorway sensors," in *Proceedings of Sensys*, 2012.

[4] S. Blackrnan and A. House, "Design and analysis of modern tracking systems," *Boston, MA: Artech House*, 1999.

[5] G. Pulford, "Taxonomy of multiple target tracking methods," in *Radar, Sonar and Navigation, IEE Proceedings-*, vol. 152, no. 5.

[6] S. Oh and S. Sastry, "Tracking on a graph," in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 26.

[7] D. De and et al, "Findinghumo: Real-time tracking of motion trajectories from anonymous binary sensing in smart environments," in *ICDCS*, 2012.

[8] G. Pulford and R. Evans, "A survey of hidden markov model tracking with emphasis on othr-first report to high frequency radar division," *University of Melbourne, Tech. Rep. CSSIP*, vol. 7, p. 95, 1995.

[9] D. H. Wilson and C. Atkeson, "Simultaneous tracking and activity recognition (star) using many anonymous, binary sensors," in *Pervasive computing*. Springer, 2005, pp. 62–79.

[10] F. Kruger, M. Kasparick, T. Mundt, and T. Kirste, "Where are my colleagues and why? tracking multiple persons in indoor environments," in *Intelligent Environments (IE), International Conference on*, 2014.

[11] A. Muller, Sebastian Hein, "Multi-target data association in binary sensor networks," in *Proceedings of the 5th International Conference on Ambient Computing, Applications, Services and Technologies*, 2015.

[12] S. Lee, D. Ahn, S. Lee, R. Ha, and H. Cha, "Personalized energy auditor: Estimating personal electricity usage," in *Pervasive Computing and Communications, 2014 IEEE International Conference on*, 2014.

[13] N. Nasir et al., "Fusing sensors for occupancy sensing in smart buildings," in *Distributed Computing and Internet Technology*. Springer, 2015, pp. 73–92.

[14] D. B. Reid, "An algorithm for tracking multiple targets," *Automatic Control, IEEE Transactions on*, vol. 24, no. 6, pp. 843–854, 1979.

[15] J. Sherrah, B. Ristic, and D. Kamenetsky, "A pedestrian multiple hypothesis tracker fusing head and body detections," in *DICTA*, 2013.

[16] J. Faghmous et al., "Multiple hypothesis object tracking for unsupervised self-learning: An ocean eddy tracking application." in *AAAI*, 2013.

[17] T. Schmitt *et al.*, "Watch their moves: Applying probabilistic multiple object tracking to autonomous robot soccer," in *AAAI/IAAI*, 2002.

[18] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 19, no. 1, pp. 5–18, 2004.

[19] D. B. Chelton, M. G. Schlax, and R. M. Samelson, "Global observations of nonlinear mesoscale eddies," *Progress in Oceanography*, vol. 91, no. 2, pp. 167–216, 2011.

[20] L. J. Cox and S. L. Hingorani, "An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, no. 2, pp. 138–150, 1996.

[21] A. Kalyanaraman and K. Whitehouse, "An event-based data fusion algorithm for smart cities," in *Proc. of 1st International Workshop on Smart Cities: People, Technology and Data*, 2015.