Fusing RFID and Computer Vision for Fine-Grained Object Tracking

Chunhui Duan*, Xing Rao*, Lei Yang† and Yunhao Liu*
*School of Software and TNLIST, Tsinghua University, China
†Department of Computing, The Hong Kong Polytechnic University, Hong Kong
Email: {hui, raoxing, young}@tagsys.org, yunhao@greenorbs.com

***Abstract*—In recent years, both the RFID and computer vision technologies have been widely employed in indoor scenarios aimed at different goals while faced with respective limitations. For example, the RFID-based EAS system is useful in quickly identifying tagged objects but the accompanying false alarm problem is troublesome and hard to tackle with except that the accurate trajectory of the target tag can be easily acquired. On the other side, the CV system performs fairly well in tracking multiple moving objects precisely while finding it difficult to screen out the specific target among them. To overcome the above limitations, we present TagVision, a hybrid RFID and computer vision system for fine-grained localization and tracking of tagged objects. A fusion algorithm is proposed to organically combine the position information given by the CV subsystem, and phase data output by the RFID subsystem. In addition, we employ the probabilistic model to eliminate the measurement error caused by thermal noise and device diversity. We have implemented TagVision with COTS camera and RFID devices and evaluated it extensively in our lab environment. Experimental results show that TagVision can achieve 98% blob matching accuracy and $10.33mm$ location tracking precision.**

***Keywords*—RFID; computer vision; tracking; TagVision**

I. Introduction

Being regarded as one of the most promising technologies of this century, Radio Frequency IDentification (RFID) has got broad applications in everyday scenarios, *i.e.*, warehouse inventorying, baggage sortation in the airport, logistics tracking, books ordering in the library and so on [1], [2]. A recent usage of RFID technology is in the Electronic Article Surveillance (EAS) system. According to an official statement last year, the total loss of commodities in retail industry has reached up to 123.4 billion dollars all over the world, accounting for 1.23% of the gross income, and unfortunately, 77% of the loss results from pilferage. To prevent goods from being stolen, EAS, which was initially designed to serve for the clothing industry, has been applied to many department stores, supermarkets, libraries, *etc.*

Compared to a conventional acoustomagnetic-based system, the RF-based EAS adopting UHF RFID devices have been well appreciated for its light volume, wireless communication, automatic identification and low cost. However, limited by its principle, existing RFID-based EAS systems are suffering from the drawback of *false alarm*, which degrades the performance dramatically. Here, the false alarm is defined as the false positive unintentionally reported by the system when users are not actually leaving or entering the warning zone.

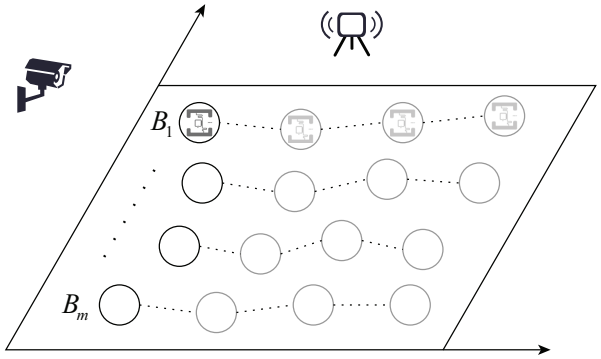


Fig. 1: Illustration of TagVision

False alarm not only causes potential trouble for the administrative staff, but also generates conflict between consumers and businesses. A similar problem also emerges in the warehouse and logistics management domain. Although the RFID-based technique wins in its quick identification speed, centralized management, security, *etc.*, when pitted against manual operation, the misreading problem (misreading the goods when they are close to the detection zone instead of passing through it) that comes along can cause information errors, impact management activity, and even result in unnecessary losses. So to summarize, the false alarm (reading) is a common issue in many RFID-based applications and how to reduce it is of significant importance.

One typical approach to solve this is to design a specialized antenna with the radiation lobe as narrow as possible, so that the percentage of unwanted reads can be minimized, however this can be relatively hard to implement. Alternatively, if we can obtain the tag's accurate location at every instant, it can help to make better decisions. But, tag localization itself is a challenging task especially in passive backscatter systems.

On the other hand, recent years have witnessed the Computer Vision (CV) technology progress to a remarkable state where reliable tracking of individual objects or people from video segments can be achieved at low overhead. Besides, the camera device is inexpensive and has wide deployment in everyday life. We can acquire comprehensive monitoring images from the region of interest with only one small predeployed properly placed camera. However, there are also practical problems bothering the CV system, *e.g.*, an item

978-1-5090-5336-0/17/$31.00 ©2017 IEEE

is found stolen in the site but there may be multiple people captured by the monitoring camera, making it hard to identify the thief. If an RFID tag is pre-attached onto the item, we can traceback the target by figuring out which trajectory is most likely to be that of the tag's.

Based on all the above, this work proposes TagVision, a fine-grained identifying and tracking schema for tagged objects, which fuses RFID and CV together using only one Commercial Off-The-Shelf (COTS) camera and RFID antenna. Fig. 1 illustrates a toy example of the scene. There exist multiple moving objects (also referred to as motion blobs, $B_1, B_2, \cdots$), among which one has the RFID tag attached while the others do not. The basic idea of TagVision is as follows. The reader continuously interrogates tags in the surveillance region during which the camera keeps monitoring moving objects in the scene. TagVision first schedules the CV subsystem to give accurate trajectory tracking results of all current motion blobs. Then for the detected RFID tag, TagVision tries to match it to one of the motion blob that is most likely to be the tag's host[1]. For this task, a proposed *matching score* will be calculated and assigned to each motion blob integrating the target tag's phase information reported by the reader from the RFID subsystem. Finally, the tag will be matched to the specific motion blob with the highest matching score. Meanwhile, the accurate trajectory of the target tag can be acquired as the corresponding tracking results from the CV subsystem.

Contributions: In summary, this paper makes the following contributions:

- First, TagVision makes an innovative combination of the computer vision and RFID technologies to enable highly accurate tracking of tagged objects, providing a new perspective to deal with practical problems in the RFID domain.
- Second, a motion blob detection and tracking mechanism is implemented, utilizing the dense optical flow method and mean-shift algorithm.
- Third, a fusion algorithm is designed to seamlessly link the location information output by the CV subsystem to the phase data collected by the RFID subsystem, which successfully handles the negative impact from phase measurement error (caused by environmental noise and tag diversity).
- Fourth, we implement the system purely with COTS camera and RFID products and conduct comprehensive evaluations. It's validated that TagVision can match the target tag to its real host with as high as 98% precision on average and track the tag at a very fine granularity.

The rest of the paper is organized as follows. The main design of TagVision is overviewed in §II. We present the technical details of TagVision in §III and §IV with regard to the two subsystems respectively. The implementation of TagVision is described in §V and evaluated in §VI. We review related work in §VII and conclude this paper in §VIII.

II. Overview

TagVision is a fine-grained passive tracking system towards tagged mobile objects, which organically combines CV to existing RFID systems. To this end, let's first look at the pros and cons of existing CV and RFID systems. CV performs well at continuously providing accurate location estimates of multiple targets while failing to differentiate them from each other. An RFID reader can efficiently identify the tags within the operating range while it is hard to estimate their accurate locations. So we can regard the CV tool as a complement to the RFID system, since it takes full advantage of both systems. In this work, we focus on locating and tracking of tags that are not moving at a high speed. The reason is that there may be severe packet loss when the tag moves rapidly, even making it unable to be read, which is beyond the scope of our paper.

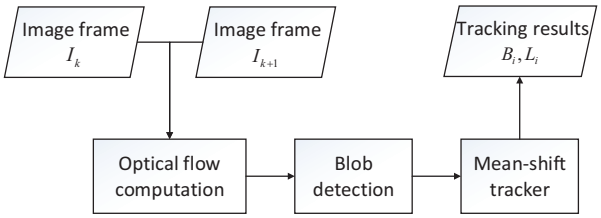


Fig. 3: Flow of motion blob detection and tracking

In the scene, there are multiple motion blobs, with one of them having an attached RFID tag. TagVision deploys a monocular camera mounted on the ceiling to provide a bird's-eye view of the surveillance region and a reader antenna to collect the phase information of the tag. Its infrastructure also includes a central server which stores the camera parameters, coordinates of the antenna and other system settings. Then, TagVision goes through the following steps at a high level to track the tagged objects:

- The camera records the image frames of the scene for a while and the reader interrogates nearby tags in the meantime. Afterwards, image data and signal snapshots are sent to the server.
- TagVision acquires the instant real-world coordinates of all the motion blobs at every frame from the image data, using the mechanism in §III.
- TagVision obtains and calibrates the phase shifts from the signal snapshots and utilizes the fusion algorithm to match the tag to one specific motion blob (see §IV).

Fig. 2 illustrates the system architecture of TagVision. The whole system is composed of two subsystems: the RFID subsystem and the CV subsystem. We will elaborate on the technical details of the above steps in the next few sections.

III. CV Subsystem

In this section, we describe how the CV subsystem works, and how it assists the RFID subsystem for object tracking. We start with the preliminary knowledge in the CV domain.

A. Detection and Tracking of Moving Objects

As an auxiliary component, the CV subsystem is designed to detect and track the moving targets in our surveillance region. After the camera captures the image frames, the blob

[1]The host refers to the blob with the target tag attached.

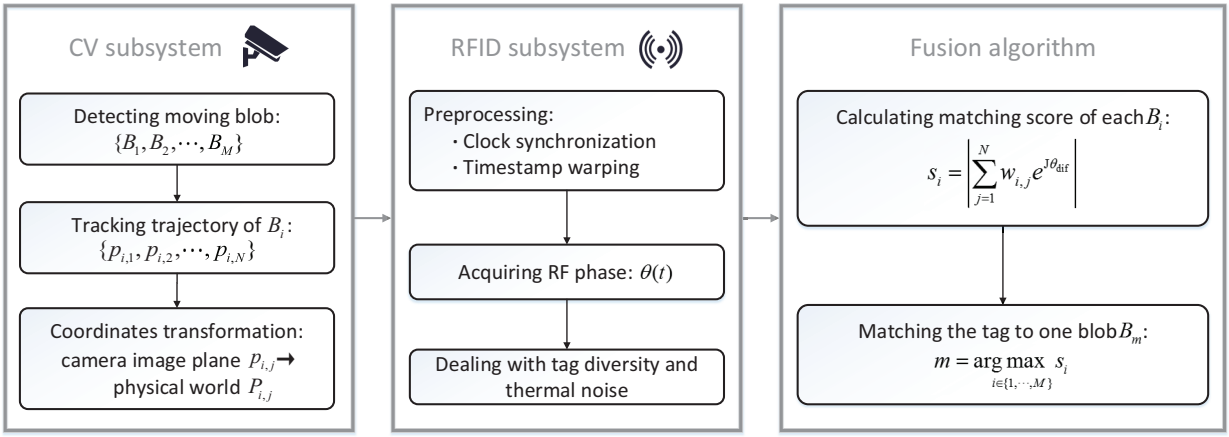


Fig. 2: System architecture

detection and tracking module goes through the steps outlined in the flowchart in Fig. 3. It relates two consecutive frames, which first go to the optical flow computation block to find the 2D-motion field. The optical flow is the apparent motion of brightness patterns in the image and in mathematical terms, if $I(x,t)$ is the image intensity at a coordinate x at time t, one wants, for each pixel x, find the displacement vector a that fulfils

$$I(x,t) = I(x+a, t+\Delta t)$$

We adopt the Lucas-Kanade optical flow method [3] to accomplish this. Then, the blob detection block is employed to detect blob instances (marked as $\mathcal{B} = \{B_1, B_2, \cdots, B_M\}$) and incorporates several constraints regarding blob size and velocity. To track the individual blobs over consecutive frames, a mean-shift tracking approach [4] is applied. After sampling a certain number of frames, we can construct the trajectory $\mathcal{L}_i$ of every moving object B_i over time, which can be represented as the two tuples of position and time, namely $\mathcal{L}_i = \{(p_{i,1}, t_1), (p_{i,2}, t_2), \cdots\}$. And $p_{i,j}$ indicates the location of blob B_i in the image at time t_j.

B. Coordinates Transformation

Up to now, the trajectories of moving objects that we get are represented as pixel values in the camera image plane. Ultimately, what we need is the trajectory in our physical world coordinate system. So the corresponding transformation between the two systems is wanted. For a better illustration, let's start with some notations in the camera model.

A 2D point in the camera image plane is denoted by $\mathbf{p} = [u, v]^{\mathrm{T}}$ and a 3D point in space is denoted by $\mathbf{P} = [X, Y, Z]^{\mathrm{T}}$.[2] We use $\tilde{\mathbf{x}}$ to indicate the augmented vector by adding 1 as the last element: $\tilde{\mathbf{p}} = [u, v, 1]^{\mathrm{T}}$ and $\tilde{\mathbf{P}} = [X, Y, Z, 1]^{\mathrm{T}}$. A camera is modeled by the usual pinhole, then the relationship between a 3D point $\mathbf{P}$ and its image projection $\mathbf{p}$ is given by

$$c\tilde{\mathbf{p}} = \mathbf{A_1}\mathbf{A_2}\tilde{\mathbf{P}} \tag{1}$$

[2] $\mathbf{A}^{\mathrm{T}}$ denotes the transpose of matrix $\mathbf{A}$.

where c is an arbitrary scale factor, $\mathbf{A_1}$ and $\mathbf{A_2}$ are called the camera intrinsic matrix and extrinsic matrix respectively. $\mathbf{A_2} = \begin{bmatrix}\mathbf{R} & \mathbf{T}\end{bmatrix}$, represents the rotation ($\mathbf{R}$) and translation ($\mathbf{T}$) which relate the world coordinate system to the camera coordinate system. And $\mathbf{A_1}$ is given by

$$\mathbf{A_1} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

with (u_0, v_0) the coordinates of the principal point, α and β the scale factors in image u and v axes, and γ the parameter describing the skewness of the two image axes.

In practice, desktop cameras usually use lens to capture more light, thus exhibiting significant lens distortion, especially radial distortion. Let (x, y) be the ideal (non-observable distortion-free) pixel image coordinates, and $(\breve{x}, \breve{y})$ the corresponding real (distorted) observed image coordinates. Then we have [5]:

$$\breve{x} = x + x\left(k_1(x^2+y^2) + k_2(x^2+y^2)^2\right)$$
$$\breve{y} = y + y\left(k_1(x^2+y^2) + k_2(x^2+y^2)^2\right)$$

where k_1 and k_2 are the coefficients of the radial distortion. The center of the radial distortion is the same as the principal point.

In order to get the real physical world coordinates of our tracking objects, we need to calibrate the camera for all the above mentioned parameters, namely intrinsic matrix $\mathbf{A_1}$, extrinsic matrix $\mathbf{A_2}$ and distortion coefficients k_1, k_2. The technique in [6] is adopted because of its simplicity and robustness. It works by letting the camera observe a planar pattern (which can be printed on a laser printer and attached to a reasonable planar surface, *e.g.*, a handbook cover) shown at a few (at least two) different orientations. Either the camera or the planar pattern can be moved by hand and the motion need not be known. Fig. 4 shows our experimental setup for camera calibration. The model plane is a 7×5 checker pattern, with every square length of $41mm$, and there are 24 corners. The camera to be calibrated is a COTS AONI D881 HD720P

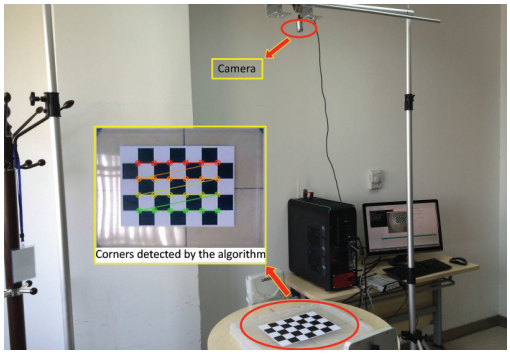


Fig. 4: Camera calibration setup

camera. We implement our calibration program using OpenCV and totally conduct 50 experiments with 24 images of the plane under different orientations taken each. Table I lists the computed results of the parameters' values. The extrinsic matrix indicating the rotation and translation of the camera can be further computed using the *solvePnP* function in OpenCV, with the intrinsic matrix and distortion coefficients as inputs.

After properly calibrating the camera, we can automatically transform the targets' location $p_{i,j}$ in the pixel image coordinate system to the real world position $P_{i,j}$. The mean error of our coordinates transformation is $1.56mm$ (see §VI-B), which is sufficient for further use.

IV. RFID Subsystem

In this section, we elaborate into TagVision's technical details in regard to the RFID subsystem. As previously described, we have realized real-time tracking of the moving objects in our surveillance region using a CV-based technique. So the problem the RFID subsystem should tackle is how to match the RFID tag to one of those trajectories.

A. Phase Model

The RF phase is a common parameter supported by commercial RFID readers. We use a reader connected with one antenna to get the tags' phase values, as illustrated in Fig. 1. Suppose $d(t)$ is the distance between the antenna and the tag at time t, the signal traverses a total distance of $2d(t)$ back and forth in backscatter communication systems. The total phase rotation output by the reader can be expressed as [7]:

$$\theta(t) = \left(\frac{2\pi}{\lambda} \times 2d(t) + \theta_{\text{div}} \right) \bmod 2\pi \tag{2}$$

where λ is the wavelength. The variable θ_{div} refers to diversity term, which is related to device hardware characteristics. The phase is a periodic function with period 2π radians which repeats every $\lambda/2$ in the distance of backscatter communication.

As the phase is directly related to the distance, to make better use of it, we should determine the geometric relationship between the reader and tag. But before more in-depth discussion, there is some preprocessing work that should be completed.

Intrinsic matrix $\mathbf{A_1}$			Distortion coefficients (k_1, k_2)
1457.2068	0	319.5	$(0.0562, 2.0569)$
0	1457.2068	239.5	
0	0	1	

TABLE I: Calibration result

B. Preprocessing

1) Clock Synchronization: Considering the CV subsystem, we get the complete trajectory of each object at total of K different time slots $\{t_1^c, t_2^c, \cdots, t_K^c\}$. Considering the RFID subsystem, we have the phase sequence reported by the reader at total L different time slots $\{t_1^r, t_2^r, \cdots, t_L^r\}$.[3] Careful reader may remark the inconsistency of the time clocks of the two subsystems, which hinders us from processing the data accurately. So the first problem we need to tackle is time synchronization between the two subsystems, *i.e.*, the computer and the RFID reader. We choose the Network Time Protocol (NTP) on the Internet as solution. The NTP can usually provide better than one milliseconds accuracy in local area networks. Our idea is to make both the computer and reader connect to the same NTP server "asia.pool.ntp.org". To achieve sufficient synchronization precision, we should config the reader online for an adequate amount of time so that it can communicate with the NTP sever adequately.

2) Timestamp Warping: Although the time is synchronized, there are gaps between the interrogation rate of the reader ($\sim$ 60 samples per second) and the frame rate of the camera ($16fps$ we adopt). Consequently timestamp sampling in the two subsystems may not align with each other. To overcome this, we refer the CV subsystem (with lower sample rate) as the benchmark, and for an arbitrary time t_k^c, we find the timestamp t_l^r which is closest to t_k^c as the corresponding timestamp in the RFID subsystem. t_l^r is formulized as below:

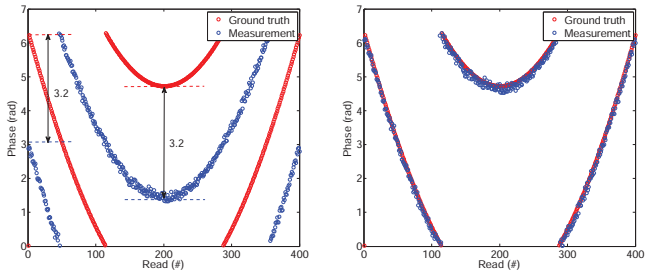
$$t_l^r = \underset{t_i^r \in \{t_1^r, t_2^r, \cdots, t_L^r\}}{\arg\min} |t_i^r - t_k^c| \tag{3}$$

In consideration of the computation and time costs, it is not necessary to deal with every collected sample, therefore assume N random snapshots of the raw data are taken. With both the clock synchronization and timestamp warping problems solved, for every moving object B_i $(i = 1, \cdots, M)$, at each time snapshot t_j $(i = 1, \cdots, N)$, we can get its location $P_{i,j}$ and the corresponding phase value θ_j of the tag. Here note that we use the same time symbol t_j in the two subsystems for convenience.

C. Fusion Algorithm

As we mentioned earlier, the key to the RFID subsystem lies in the geometric relationship between the reader and target tag, which in turn can be inferred from the tracking results output by the CV subsystem. The fusion algorithm is designed

[3] t^c and t^r represent the time clocks in the CV and RFID subsystems respectively.



(a) Original phase (b) Calibrated phase

Fig. 5: Misalignment of phase measurements

to tactfully associate the two types of data from the two subsystems.

The antenna is denoted as A with coordinates of (x_A, y_A, z_A). Let $\vartheta_{i,j}$ be the theoretical phase value of the i^{th} object at time t_j. Ignoring the diversity term, $\vartheta_{i,j}$ can be calculated as follows:

$$\vartheta_{i,j} = \frac{4\pi}{\lambda}|P_{i,j}A| \bmod 2\pi \tag{4}$$

where $|\cdot|$ measures the Euclidean distance between position P and A. And

$$|P_{i,j}A| = \sqrt{(x_{i,j}-x_A)^2 + (y_{i,j}-y_A)^2 + (z_{i,j}-z_A)^2} \tag{5}$$

where $(x_{i,j}, y_{i,j}, z_{i,j})$ denotes the space coordinates of $P_{i,j}$.

As we know, tag's phase rotation output by the reader is associated with the diversity factor θ_{div}. So there exists misalignment between the measured phase and theoretical one. An empirical study is conducted to get a better understanding. We attach a tag onto a toy train and make it move with a uniform speed of $7cm/s$ along a linear track $x = 10cm$ $(-70cm \leq y \leq 70cm)$ while the reader locates at $(120cm, 0)$. Fig. 5(a) shows the collected phase shifts and ground truth. The ground truth is calculated using Eqn. 4. Through comparison, we find that there is about 3.2 radians misalignment between them, resulting from θ_{div}. Since the misalignment remains relatively unchanged under the same macro environment (*e.g.*, same temperature, humidity, *etc.*), it is reasonable to assume θ_{div} is a constant term in the obtained phase sequence. Then we can use the first phase value as a reference to eliminate the misalignment's influence as

$$\theta(t_i) - \theta(t_1) = (\vartheta(t_i) + \theta_{\text{div}}) - (\vartheta(t_1) + \theta_{\text{div}}) = \vartheta(t_i) - \vartheta(t_1) \tag{6}$$

where the symbol ϑ denotes the corresponding theoretical phase value. The term θ_{div} is removed by doing so. Fig. 5(b) shows the results after calibrating for the device diversity. Both of the sequences synchronize well with each other.

We propose the fusion algorithm which works by matching the obtained phase sequence to each of the motion blobs (referred as blob matching) through assigning a *matching score* to each trajectory. The whole procedure is summarized in Algorithm 1. The matching score represents the possibility that the corresponding object may be the tag's host, and a higher matching score indicates the tag is more likely to be on that track. Then the problem is how to define the matching score. It's known that tag's phase measurement results contain random errors, which follow a typical Gaussian distribution with a standard deviation of 0.1 radians, *i.e.*, $\theta - \vartheta \sim \mathcal{N}(0, 0.1)$ [1]. Thus we should consider the measured phase as a Gaussian random variable instead of a constant value. Based on this, a probabilistic model is employed to define the matching score s_i of the i^{th} motion blob which is given by:

$$s_i = \left| \sum_{j=1}^{N} w_{i,j} e^{\mathbf{J}\theta_{\text{ref}}} \right| \tag{7}$$

where

$$\begin{cases} w_{i,j} = f(\theta_{\text{ref}}; 0, 0.1 \times \sqrt{2}) \\ \theta_{\text{ref}} = (\theta_j - \theta_1) - (\vartheta_{i,j} - \vartheta_{i,1}) \\ f(x; \mu, \sigma) = \dfrac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \end{cases}$$

$f(x; \mu, \sigma)$ is the Probability Density Function (PDF) of Gaussian distribution $\mathcal{N}(\mu, \sigma)$ and $\vartheta_{i,j}$ can be calculated with Eqn. 4 and Eqn. 5.

The term $\mathbf{J}$ denotes the imaginary unit and we use the complex number $e^{\mathbf{J}\theta}$ to express the wireless signal with unit amplitude. Notice that $\theta_{\text{ref}} = (\theta_j - \theta_1) - (\vartheta_{i,j} - \vartheta_{i,1}) = (\theta_j - \vartheta_{i,j}) - (\theta_1 - \vartheta_{i,1})$. Suppose the tag is on the i^{th} motion blob, then $(\theta_j - \vartheta_{i,j}) \sim \mathcal{N}(0, 0.1)$ and $(\theta_1 - \vartheta_{i,1}) \sim \mathcal{N}(0, 0.1)$, so $\theta_{\text{ref}} \sim \mathcal{N}(0, 0.1 \times \sqrt{2})$. The assigned weight $w_{i,j}$ represents the probability that the measured phase is emitted from A and backscattered at the i^{th} blob at the time t_j. By doing so, the matching score is enhanced for objects with a higher possibility to be the ground truth and weakened for others. If the tag is factually on blob B_i, then the theoretical phase should equal the measured one. The vector of $e^{\mathbf{J}\theta_{\text{ref}}}$ will get close to the real axis and $w_{i,j}$ will reach its maximum value because θ_{ref} approaches 0. All observations from different

Algorithm 1 Fusion algorithm

Input:
Trajectories of all M motion blobs: $\{\mathcal{L}_1, \mathcal{L}_2, \cdots, \mathcal{L}_M\}$; Tag's phase sequence: $\theta(t)$; Number of sampling points: N; Number of iterations: I;

Output:
The tag's host B_m;

1: Randomly sample N time snapshots $t_1, t_2, \cdots, t_N$;;
2: **for** each $i \in [1, M]$ **do**
3: Get corresponding N locations $P_{i,1}, P_{i,2}, \cdots, P_{i,N}$ from $\mathcal{L}_i$;
4: Get N sampled phase $\{\theta_1, \theta_2, \cdots, \theta_N\}$ from $\theta(t)$;
5: Calculate the matching score s_i of blob B_i;
6: **end for**
7: Repeat step 1-6 for I times and average for every s_i;
8: $m \leftarrow \arg\max s_i, i \in \{1, 2, \cdots, M\}$;
9: **return** B_m;

signal snapshots add up for each other. Otherwise, if B_i is not the ground truth, there will be deviation between the theoretical and measured phase. Then $w_{i,j}$ will get a smaller value and different signal observations cancel each other out, leading the final superimposition of all snapshots to a much lower level.

For every moving object B_i detected in the CV subsystem, we can calculate a corresponding matching score s_i using the aforementioned fusion algorithm. Further, to avoid contingency and enhance the robustness of our method, we run the fusion algorithm repeatedly for several iterations with data sampled randomly each time and average the results. Eventually, the tag can be matched to one specific moving object with the highest matching score:

$$m = \underset{i \in \{1,2,\cdots,M\}}{\arg\max}\, s_i \tag{8}$$

B_m is then output as the target. The trajectory $\mathcal{L}_m$ of the tag is also retrieved simultaneously from the CV subsystem. That is the completed workflow of our method. In more complex scenario, multiple objects may be stolen at the same time. Intuitively, we can repeat the above matching procedure for each identified tag and use more antennas to reduce the influence of multi-path, which forms a part of our future work.

D. Simulation Results

To demonstrate the feasibility of our approach, a typical indoor scenario is simulated as depicted in Fig. 6: total 101 linear tracks are generated along the x-axis at an interval of $6mm$, namely $X_1 = -300, X_2 = -294, \cdots, X_{101} = 300$. The ground truth is $X_{51} = 0$ marked with red and the reader locates at $(1018mm, 0)$. Here for the sake of expression, we consider the trajectories on 2D plane. For each iteration, we randomly select 30 sampling points within the y-axis range of $[-300, 300]$ on every trajectory and calculate the corresponding matching score using our fusion algorithm. The phase value adopted in our simulation is generated by adding White Gaussian Noise to the theoretical value. Fig. 6(b) and Fig. 6(c) are the averaged results of one iteration and 100 iterations respectively. The matching score is normalized using the maximum value as reference. We can clearly see that in either of the situations, the ground truth has an overwhelmingly higher value when compared to other tracks. Besides, the result in Fig. 6(c) is more uniform than that in Fig. 6(b). This is because one-time sampling may introduce uncertainty to the result, but through a heavy amount of sampling and taking the average, some biased values can be reduced down, making the result more convincing. As can be seen, while the real trajectory of the tag has a maximum matching score of 1, the amplitudes of other tracks are far smaller, almost all below 0.075. This validates the effectiveness of our approach.

V. Implementation

We build a prototype of TagVision using a COTS camera and UHF RFID devices.

Hardware: In the CV subsystem, we use an AONI D881 HD720P camera [8] which can support up to $60fps$ frame rate.

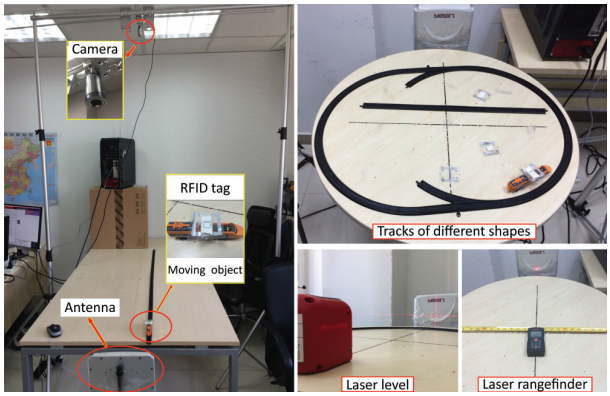


Fig. 7: Experiment setup

The camera is cheap and only costs about 100 CNY. In the RFID subsystem, we adopt an ImpinJ Speedway Revolution R420 reader [9] which is compatible with EPC Gen2 standard and operates during the frequency band of $920.5 \sim 924.5$ MHz by default. We only employ one antenna with circular polarization manufactured by Yeon technology [10], whose size is $225mm \times 225mm \times 40mm$. The reader is connected to our host end through wireless network. Two types of tags from Alien Corp [11], modeled "2×2" with size of $44 \times 46mm^2$ and "Squig" with size of $44.5 \times 10.4mm^2$ are employed. Both of them are low-cost (only about 5 cents per tag on average).

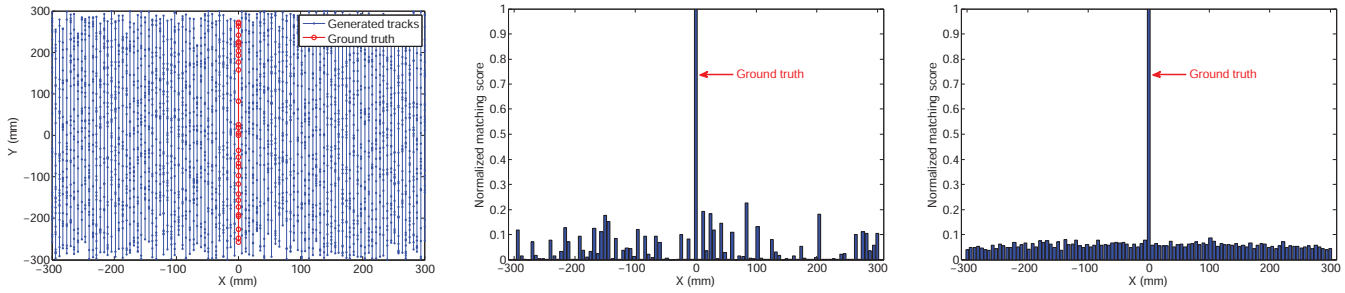
Software: We implement our algorithm including camera calibration and object tracking in CV subsystem using OpenCV computer vision library with C/C++ language. For the RFID subsystem, we adopt LLRP (Low Level Reader Protocol) [7] to communicate with the reader. ImpinJ reader extends this protocol for supporting the phase report. We adjust the configuration of reader to immediately report its readings whenever tag is detected. The client code is implemented using Java language. We have also noticed and solved the time synchronization problem between the two subsystems' time clocks as mentioned before in §IV-B. We use a Dell PC to run our all our program, as well as connecting to the reader under LLRP. The machine equips Intel Core i5-4440 CPU at 3.1GHz and 8GB memory.

VI. Evaluation

In this section, we conduct performance evaluation of TagVision in our lab environment as shown in Fig. 7.

A. Evaluation Methodology

Three groups of experiments are designed and performed in an office room whose size is $400 \times 800cm^2$. We emulate a mobile object via a toy train on which a tag is attached if needed, moving on tracks of different shapes (linear or arc-shaped). The location of the antenna is measured by a laser rangefinder with an error of $\pm 0.1mm$ under the support of a laser level. We adopt the *error distance*, defined as the Euclidean distance between the result and ground truth, as our basis metric.



(a) Tracks generated (b) Matching score with 1 iteration (c) Matching score with 100 iterations

Fig. 6: Simulation results of the fusion algorithm. (a) The 101 generated tracks. (b) The calculated matching score with one iteration. (c) The calculated matching score with 100 iterations.

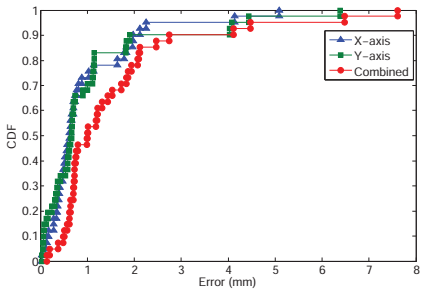


Fig. 8: Error of coordinate transformation

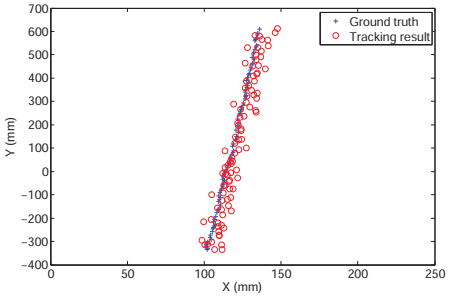


Fig. 9: Comparison of tracking result and ground truth

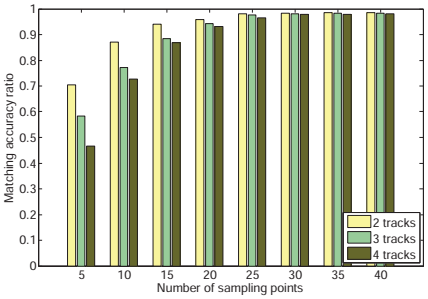


Fig. 10: Matching accuracy ratio *vs.* sampling number

B. Accuracy of Coordinate Transformation

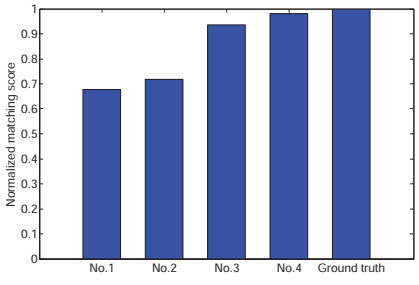
As a core model of the CV subsystem, the accuracy of coordinate transformation between the camera image plane and the physical world has a direct influence on the final results of our method. After calibrating the camera for necessary parameters as we mention in §III-B, we carry on experiments on a table top to inspect the accuracy. We establish the Cartesian coordinate system with regard to the desktop, then select and mark 41 different points uniformly scattering among the desktop plane as the ground truth, whose locations have been measured in advance. Fig. 8 shows the CDF of coordinate transformation error. The mean error distance is $0.96mm$ in x-axis, $1.05mm$ in y-axis and $1.56mm$ in combined dimension with standard deviation of $1.57mm$. Besides, 90% of the errors are less than $2.74mm$ with minimal error of $0.14mm$ and maximal error of $7.61mm$. Overall, the accuracy of coordinate transformation is very high and lays the foundation for our latter work.

C. Accuracy of Object Tracking

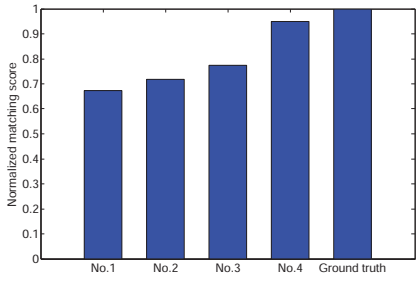
In order to achieve better result for tracking, TagVision adopt the dense optical flow method to detect the moving blob and combine the mean-shift tracking algorithm. To assess the performance, we lower the frame rate of the video and get the ground truth by manually marking the center location of the toy train at every frame. As illustrated in Fig. 9, the ground truth is marked as '+' with blue and the tracking result output by our CV subsystem is noted as '$\circ$' with red. On the basis of 20 repeated experiments, the mean error distance of our tracking method is $3.94mm$ in x-axis, $3.25mm$ in y-axis and $5.68mm$ in combined dimension with standard deviation of $4.79mm$, which is small enough compared to the size of the toy train, about $21 \times 182mm^2$.

D. Performance of Trajectory Matching

1) Matching Accuracy: Playing the role of being a bridge between the CV subsystem and RFID subsystem, the blob matching module's performance is of significant importance to TagVision. To evaluate it, we conduct the following experiment. We deploy four tracks with each about $20cm$ apart from another, as depicted in Fig. 11. The RFID tag is placed on the blob at Track 4. We run our algorithm to calculate the matching score of each track. Fig. 12 reveals the normalized results of iterating for 1 and 20 times respectively. The ground truth is calculated by manually marking the moving blob with the RFID tag attached as mentioned earlier. And we randomly sample 30 positions from the track at each time of iteration. It can be clearly seen from the figure that in both of the cases, the Track 4 gains the highest matching score, which is consistent with the ground truth because the RFID tag is indeed on the forth blob. But, with only 1 iteration, the matching score of Track 4 is very close to that of Track 3 and not distinguishable enough compared to other tracks, while the matching scores become more stable and easy to tell when we increase the iteration time to 20, or even more. Our experimentation shows



(a) Iteration: 1



(b) Iteration: 20

Fig. 12: Comparison of different tracks' matching scores under different iteration times. (a) Iterating once. (b) Iterating for 20 times.

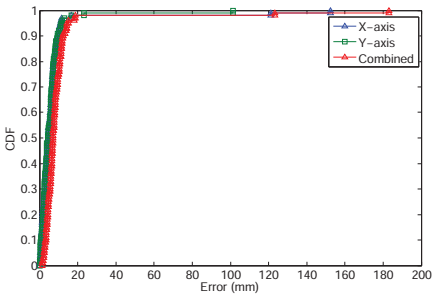


Fig. 13: Overall tracking error

that 20 iterations are sufficient to make the result robust and discriminative and it only takes $0.028s$ to finish, so we set the default number of iterations to 20. Besides, since our CV subsystem has relatively high precision, the calculated matching score of Track 4 approximates the ground truth very much.

Furthermore, we vary the number of tracks from 2 to 4 with different shapes (linear and non-linear) to make a comparison study. We define the *matching accuracy ratio* R as below

$$R = \frac{\#\text{ of successful matches}}{\#\text{ of experiments in total}} \times 100\% \tag{9}$$

Table II depicts the obtained accuracy ratio along with number of tracks. It's obvious that all of the cases get very high accuracy ratio (98% on average) and there isn't much difference among them (less than 1% gap). This demonstrates the effectiveness of our fusion algorithm. Besides, with the number of tracks increasing, the accuracy ratio also rises a little. This is reasonable because the fewer tracks, the less error will be accumulated.

2) Impact of Sampling Number: The output trajectory from the CV subsystem usually contains more than 100 positions each and we randomly sample 30 by default. So we wonder whether the number of sampling points will have an impact on the final blob matching accuracy. To study this, we range the sampling number from 5 to 40 at an interval of 5, and for each case, we test on 2, 3 and 4 tracks respectively. Note that we adopt the default number of iterations 20. Fig. 10 describes how the matching accuracy ratio changes along with the sampling number. We have the following observations.

- With the number of sampling points increases, the accuracy ratio also exhibits an apparently rising pattern, from 59% when sampling number is 5 to 98% when sampling number reaches 40. The reason why the accuracy becomes relatively low when the sampling number is less than 15 is that the accumulated points are too limited in the fusion algorithm, thus the error incurred by environmental noise will cause greater influence on the final result, making the target track more difficult to be differentiated.
- The matching accuracy remains stable at 98% when sampling number arrives at a certain level (≥ 30). This is because the algorithm itself becomes the bottleneck when we sample adequately. The accuracy already reaches up to 94% when sampling 20 points. To gain a tradeoff between computation cost and precision, we choose 30 as the default sampling number.
- Under the same sampling number, the fewer tracks there are, the higher precision can be achieved, which is consistent with our conclusion in the previous subsection.

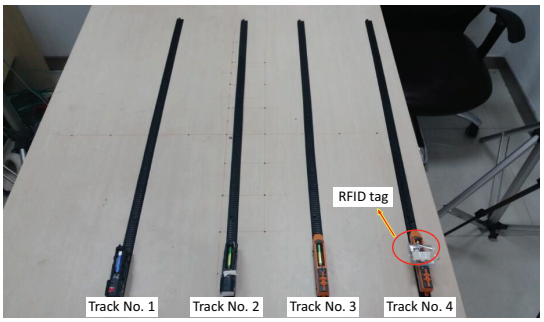


Fig. 11: Four tracks scene

E. Overall Accuracy of Tag Tracking

As TagVision aims at tracking tagged object, the final tracking accuracy of the target tag is the most crucial metric. Fig. 13 plots the CDF of tracking error. We observe that most of the errors are within $10mm$ as the tag is correctly matched to its host except very few outliers beyond $100mm$, which are caused by the blob matching fault. The mean error distance is $7.55mm$ in x-axis, $5.91mm$ in y-axis, and $10.33mm$ in combined dimension with standard deviation of $21.23mm$, and 90% of the errors are less than $12.13mm$, which is good enough for many applications. It is the excellent performance

# of tracks	2	3	4
Accuracy ratio R	98%	98%	97%

TABLE II: Matching accuracy

of the fusion algorithm that guarantees the overall tracking accuracy of tagged object.

VII. Related Work

Fine-grained localization and object tracking have been well studied in both CV and RFID field.

CV-based techniques: Recent years have witnessed the rapid advance of computer vision, [6], [12], making it possible for reliable tracking of individual objects or people [13]. Common visual features for tracking include color, edges, optical flow, texture, *etc.*, or a combination of these for better performance. Comaniciu *et al.* [4] design a method for real-time tracking of non-rigid objects based on the mean-shift iterations. The authors in [14] propose a robust object tracking algorithm using a collaborative model. Zhang *et al.* [15] presents a new multi-object model-free tracker that can revolve the problem of similar appearance. It is the maturity of CV technology that motivates us and establishes the basis of our method.

RFID-based techniques: Localization utilizing RF signal has drawn the attention of many researchers. Early works rely on RSSI information as the fingerprint or distance ranging metric to acquire location information [16], [17]. There is also growing interest in utilizing phase information to locate tags. Angle of Arrival (AoA) is a typical solution, which works by measuring the phase difference between the received signals at different antennas [18], [19]. BackPos [20] proposes the hyperbolic positioning using the phases detected by antennas under triangle constraint. Tagspin [21] makes an innovative improvement to the traditional AoA approach and first quantifies the tag orientation's impact. Synthetic Aperture Radar (SAR) is another technique taking advantage of moving antenna or tag to simulate antenna array for hardware cost consideration [22], [23]. PinIt [24] captures and extracts multi-path profiles via SAR to locate RFIDs. Tagoram [1] realizes real-time tracking of mobile tag to a very high precision.

There is also some literature following on the combination of CV and RFID systems for localization and tracking. Goller *et al.* [25] integrate the information from the two sensor modalities in a probabilistic manner, providing robustness to false positive observations. The authors in [26] have investigated on a camera-assisted RFID localization technique for passive UHF label, leading to substantial improvement on accuracy. However, both of the works utilize the RSSI for model construction, which is proven to be unreliable because of its high sensitivity to multi-path propagation and antenna gain, thus difficult to achieve high precision.

VIII. Conclusion

In this work we present a tagged object identifying and tracking system based on the fusion of RFID's strengths and CV's localization capability. We implement a prototype of TagVision with COTS devices and conduct extensive evaluations. The results show that TagVision can achieve fairly good accuracy with strong robustness. We believe with the tag being tracked at a fine granularity, our system will open up new opportunities in practical deployments.

Acknowledgment

This study is supported in part by NSF China Grant No. 61572282 and NSFC/RGC Joint Research Scheme No. 61361166009.

References

[1] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu, "Tagoram: Real-time tracking of mobile rfid tags to high precision using cots devices," in *Proceedings of ACM MobiCom*, 2014.
[2] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu, "Relative localization of rfid tags using spatial-temporal phase profiling," in *Proceedings of USENIX NSDI*, 2015.
[3] A. M. Tekalp, *Digital Video Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1995.
[4] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings of IEEE CVPR*, 2000.
[5] G.-Q. Wei and S. D. Ma, "Implicit and explicit camera calibration: Theory and experiments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 469–480, 1994.
[6] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.
[7] Impinj, "Speedway revolution reader application note: Low level user data support," in *Speedway Revolution Reader Application Note*, 2010.
[8] "AONI," http://www.anc.cn/.
[9] "Impinj, Inc," http://www.impinj.com/.
[10] "Yeon Antenna," http://www.yeon.com.tw/content/product.php?act=detail&c_id=43.
[11] "Alien," http://www.alientechnology.com/tags/square.
[12] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," in *Proceedings of IEEE CVPR*, 2004.
[13] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006.
[14] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Proceedings of IEEE CVPR*, 2012.
[15] L. Zhang and L. van der Maaten, "Structure preserving object tracking," in *Proceedings of IEEE CVPR*, 2013.
[16] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: Indoor location sensing using active rfid," *Wireless Networks*, vol. 10, no. 6, pp. 701–710, 2004.
[17] L. Shangguan, Z. Li, Z. Yang, M. Li, and Y. Liu, "Otrack: Order tracking for luggage in mobile rfid systems," in *Proceedings of IEEE INFOCOM*, 2013.
[18] P. Nikitin, R. Martinez, S. Ramamurthy, H. Leland, G. Spiess, and K. Rao, "Phase based spatial identification of uhf rfid tags," in *Proceedings of IEEE RFID*, 2010.
[19] S. Azzouzi, M. Cremer, U. Dettmar, R. Kronberger, and T. Knie, "New measurement results for the localization of uhf rfid transponders using an angle of arrival (aoa) approach," in *Proceedings of IEEE RFID*, 2011.
[20] T. Liu, L. Yang, Q. Lin, Y. Guo, and Y. Liu, "Anchor-free backscatter positioning for rfid tags with high accuracy," in *Proceedings of IEEE INFOCOM*, 2014.
[21] C. Duan, L. Yang, and Y. Liu, "Accurate spatial calibration of rfid antennas via spinning tags," in *Proceedings of IEEE ICDCS*, 2016.
[22] J. Wang, F. Adib, R. Knepper, D. Katabi, and D. Rus, "Rf-compass: Robot object manipulation using rfids," in *Proceedings of ACM MobiCom*, 2013.
[23] J. Wang, D. Vasisht, and D. Katabi, "Rf-idraw: Virtual touch screen in the air using rf signals," in *Proceedings of ACM SIGCOMM*, 2014.
[24] J. Wang and D. Katabi, "Dude, where's my card?: Rfid positioning that works with multipath and non-line of sight," in *Proceedings of ACM SIGCOMM*, 2013.
[25] M. Goller, C. Feichtenhofer, and A. Pinz, "Fusing rfid and computer vision for probabilistic tag localization," in *Proceedings of IEEE RFID*, 2014.
[26] T. Nick, S. Cordes, J. Gotze, and W. John, "Camera-assisted localization of passive rfid labels," in *Proceedings of IEEE IPIN*, 2012.