International Conference on Artificial Reality and Telexistence
Eurographics Symposium on Virtual Environments (2019)
Y. Kakehi and A. Hiyama (Editors)

Random-Forest-Based Initializer for Real-time Optimization-based 3D Motion Tracking Problems

Jiawei Huang[1] Ryo Sugawara[1] Taku Komura[2] Yoshifumi Kitamura [1]

[1]Research Institute of Electric Communication, Tohoku University
[2]Edinburgh University

Abstract

Many motion tracking systems require solving inverse problem to compute the tracking result from original sensor measurements, such as images from cameras and signals from receivers. For real-time motion tracking, such typical solutions as the Gauss-Newton method for solving their inverse problems need an initial value to optimize the cost function through iterations. A powerful initializer is crucial to generate a proper initial value for every time instance and, for achieving continuous accurate tracking without errors and rapid tracking recovery even when it is temporally interrupted. An improper initial value easily causes optimization divergence, and cannot always lead to reasonable solutions. Therefore, we propose a new initializer based on random-forest to obtain proper initial values for efficient real-time inverse problem computation. Our method trains a random-forest model with varied massive inputs and corresponding outputs and uses it as an initializer for runtime optimization. As an instance, we apply our initializer to IM3D, which is a real-time magnetic 3D motion tracking system with multiple tiny, identifiable, wireless, occlusion-free passive markers (LC coils). During run-time, a proper initial value is obtained from the initializer based on sensor measurements, and the system computes each position of the actual markers and poses by solving the inverse problem through an optimization process in real-time. We conduct four experiments to evaluate reliability and performance of the initializer. Compared with traditional or naive initializers (i.e., using a static value or random values), our results show that our proposed method provides recovery from tracking loss in a wider range of tracking space, and the entire process (initialization and optimization) can run in real-time.

CCS Concepts

• ***Computing methodologies*** → ***Classification and regression trees; Motion capture;***

1. Introduction

Over the decades motion tracking systems have been dramatically improved and are being widely used. Despite differences in tracking principles, a key process for many of such systems is optimization to solve the inverse problem. A typical optimization approach is the Gauss-Newton method. Starting from an initial value, it minimizes the mean square error of a cost function through iterations. Due to the nature of the solver, a superior initializer that generates a proper initial value, which leads to the solution, is required to allow the optimization process to converge at the correct solution for the inverse problem. Without proper initial value, the result of the optimization process is inaccurate or even wrong. Thus, developing a superior initializer is critical and an effective way to improve the motion tracking quality. Even though many new motion tracking systems have been proposed, the initializer issues have not been adequately discussed. Consequently, such systems continue to use conventional or problem-specific initializers. Actually, many motion tracking systems require that a specific initializer be developed for their own problem through experimental processes, e.g. testing and customizing multiple initializers. This situation generally exists in optical motion tracking systems (e.g., [PK07]) and magnetic motion tracking systems (e.g., [RBSJ79]).

The popularity of machine-learning methods (e.g., [Bis06]) continues to grow, and they are widely being used in 3D motion tracking, too. Unlike other conventional initializer (e.g., random guesses), machine-learning methods predict output values from current input data based on training data: collected samples. For inverse-problem-based motion tracking technologies, even though machine-learning methods cannot provide such accurate results (perhaps due to the ambiguity of such inverse problems), they can predict a close value very quickly (especially tree-based methods such as random-forest or KD-trees), which is suitable for the requirements of a fast and accurate initializer. However, this potential has not been fully investigated.

In this paper, we propose a novel random-forest-based initializer for optimization-based 3D motion tracking problems. Compared with other initializers or initialization methods, our method provides a more accurate initial value in a short computational time and can be further applied to computations of various motion

DOI: 10.2312/egve.20191273

tracking systems. With initial values close to the solution provided by this initializer, real-time 3D motion tracking systems achieve less divergence of optimization processes and faster recovery. To demonstrate our approach's benefit, we apply the initializer to the IM3D system [HTHK14] because its calculation process is one typical example of solving the inverse problem by the Gauss-Newton method whose tracking result sometimes cannot be obtained when the magnitude of the magnetic field is inadequate. Based on measured magnetic flux from sensors, it computes the spatial configuration of markers (LC coils) by the Gauss-Newton method. During run-time, it uses the previous frame's result as an initial value to provide a relatively reliable initial value, although it does not always lead to a solution when the marker is moving fast. Once the system falls into a situation that it fails to track (i.e., S/N ratio dramatically decreases when the system's marker becomes specific poses), the tracking result becomes unreliable as the initial value for the next frame, and thus the recovery fails. Hence, the tracking quality must be improved,by introducing an accurate real-time initializer.

This paper consists of four main sections. In section 3, we propose the main concept of our new initializer. In section 4, we briefly review the random-forest method and the IM3D system. In section 5, we introduce our workflow to implement our initializer in IM3D, including data collection, preprocess, and run-time implementation. We describe the configuration of the meta-parameters in the random-forest model in sections 5 and 6 and evaluate the real-time performance and discuss the benefits of our initializer in section 6.

Our main contribution is a new data-driven initializer for real-time optimization-based 3D motion tracking systems.

2. Related Work

Our work uses a machine-learning method to solve the initialization problem for motion tracking systems. Thus, our work is mainly related to 3D motion tracking systems and machine-learning applications for motion tracking.

2.1. 3D Motion Tracking System

First, we review 3D motion tracking systems that suffer from initialization problems. Most marker-based tracking systems, which compute the spatial configuration of markers by solving the inverse problem based on sensor values, commonly apply such optimization methods as the Gauss-Newton method for this task. Marker-based optical tracking systems (e.g., [PK07]) are very popular these years. They first obtain the initial value from linear singular value decomposition (SVD), and then each frame's result becomes the next frame's initial value. Even though SVD is an accurate method for initialization, its computational complexity requires huge computation resources, and consequently it cannot be used for real-time tracking systems.

Magnetic motion tracking systems (e.g., [Pol]), which are suitable for precise tracking tasks, are also hampered by this initialization problem. Although a previous work [RBSJ79] argued that this initial value is important for the convergence of the optimization process, it did not discuss how to obtain a proper one. Tracking loss occurs when the S/N ratio of the sensor value becomes too small, and a proper initial value (position and orientation) close to the actual result becomes necessary. Without a proper initial value, the computation result will not converge, and tracking recovery requires more frames.

Two other works [HTHK14] and [HMT*15] proposed another magnetic tracking approach that applies a typical Gauss-Newton method to solve the inverse problem. Similar to other approaches, these systems only start with the origin as an initial value, and each frame uses the previous frame's result as an initial value based on the expectation that the previous frame's result is acceptably close to the current frame's solution. However, these two works do not provide a solution to re-initialize the tracking once it gets lost. Consequently, the tracking suffers from an inability to recover from tracking loss.

A vision-based hand-tracking system was introduced [ZCX12] with a particle-based initialization method, which starts the optimization process from many different initial values and chooses the best result. However, such methods are not suitable for real-time use because they require a large amount of computation resources.

Even though many researches have proposed new ideas on optimization-based motion tracking techniques, they rarely discuss how to obtain proper initial values. However, we believe that this problem deserves more discussion since the optimization process of practical motion tracking systems frequently diverges and struggles to recover.

2.2. Data-Driven Methods for Motion Tracking

Data-driven methods, or machine-learning methods, can build nonlinear models. They usually adjust the parameters for specific inputs and outputs through training to output desired values for similar inputs in run-time. Popular techniques include multi-layer neural networks (e.g., deep neural network (DNN), convolutional neural networks [KSH12]), support vector machines (e.g., [CV95]), and multi-layer decision trees (e.g., [THF*03]).

Recently with the improvement of and deep research on data-driven methods, many new motion tracking techniques have appeared. For example, a system was introduced [MFWM17] that trains a DNN with Doppler sensor values as inputs to regress hand motions. Even though its accuracy is relatively low, it demonstrates the potential of multi-layer neural networks for motion tracking systems. Two other works [SFC*11] and [SKR*15] used random-forest to track full-body or hand motions from depth images, although their methods do not address the inverse problem. These researches solved the initialization problem of previous systems with high performance and robust per-frame initialization. This implies that, even though the result of the random-forest method may be less accurate than optimization, it can be applied to give proper initial values in acceptable accuracy for motion tracking systems that solve the inverse problem by optimization.

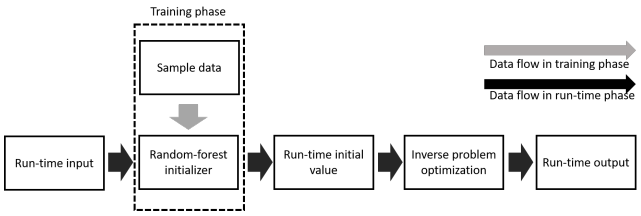


Figure 1: *Workflow of initializer*

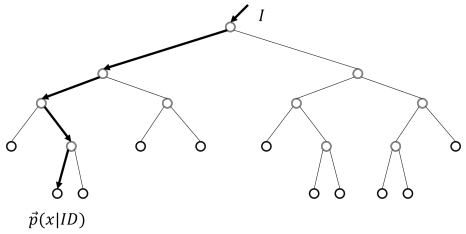


Figure 2: *Decision tree in random-forest for parameter x*

3. Data-driven Initializer for Real-time 3D Motion Tracking

3.1. Objective

As stated above, in motion tracking systems, most inverse problems are solved by the Gauss-Newton method. Denote $V_i(t)$ as an initial value and $V_r(t)$ as a result at time instance t. The computation starts from initial value $V_i(t)$ and optimizes the cost function's minimum squared error through steps and gets converged result $V_r(t)$. When error E between $V_i(t)$ and $V_r(t)$ is too large, this method fails to get a proper result. The previous frame's result $V_r(t-1)$ is usually used as the current frame's initial value $V_i(t)$. Once the tracking is lost, the error between $V_r(t)$ and $V_i(t)$ becomes too large, causing the optimization process to diverge. We define function $P(i)$, which predicts an initial value through raw input data I, and the error as

$$E = |P(i) - V_r(t)| \tag{1}$$

We need to find a better function $P'(i)$ so that most cases E can be reduced through optimization (i.e., calculation converges) and applied in real-time. In this paper we prove that for a specific problem, random-forest offers a better $P'(i)$ with less error than conventional methods and low computation resource requirements to ensure convergence of the optimization process.

3.2. Workflow

As shown in Fig. 1, the workflow that builds the initializer includes two phases: training and run-time. In the training phase, we ran a simulation of the motion tracking system to generate as many theory-based samples as possible and make input-output pairs. Then we trained the random-forest with these samples to get a classification model. In the run-time phase for every frame, we put the latest sensor data into the model to predict the output as an initial value to solve the inverse problem. This workflow, which can be applied to all categories of inverse-problem-based tracking systems that need initial values, adds a per-frame initialization to them. As stated in section 3.1, as long as the predicted initial value has less error than the other methods, the divergence of the optimization process decreases, especially when tracking loss occurs.

3.3. Random-Forest

Random-forest, or a random decision forest, is an effective multiclass classifier that consists of multiple decision trees with splits and leaf nodes (Fig. 2). Each split node consists of feature f_θ and threshold τ. To classify input set I, start at the root and repeatedly evaluate Eq. 2, branching left or right based on the comparison to threshold τ. At each leaf node in the tree, the distribution of output $\vec{P}(X|ID)$ is stored:

$$f_\theta(I,x) = d_I(x + \frac{u}{d_I(x)}) - d_I(x + \frac{x}{d_I(x)}) \tag{2}$$

The distributions are averaged for all the trees in the forest to give the final distribution, which is the final possibility of this classifier's output. A random-forest can be effectively trained with a previously described algorithm [THF*03].

Random-forest has two main configurable parameters: the depth of the trees and their number, both of which determine the model's complexity. In practice, we configure these parameters based on the model's actual performance through experiments. In section 5, we also show an experiment with which we configured the model for our application example.

3.4. Sample Acquisition

Collecting training data is sometimes difficult for precise and high-resolution motion tracking systems. In most previous researches, training data were carefully collected through actual use cases, for example, using accurate robot arms for automatic measurements or manual measurements in small intervals of positions ($< 5mm$) and rotations (10°) for the entire tracking space since the machine-learning model's output directly becomes tracking results. However, our initializer does not require such sampling because the random-forest's output is used as an initial value that only has to be accurate enough for the optimization process. This actually introduces a possible approach to simply acquire massive samples. As Eq. 3 describes, the optimization process of 3D motion tracking systems can be generalized to a process that minimizes the objective function. Here x is the tracking result, M is the measurements, and $f(x)$ calculates the theoretical measurements for specific tracking result x based on a tracking principle. Therefore, such optimization is seeking theoretical value x so that $f(x) = M$. Based on this, we use a simulator to enumerate every possible x to compute corresponding $f(x)$ and use these combinations as training samples:

$$E(x) = |M - f(x)| \rightarrow Minimum \tag{3}$$

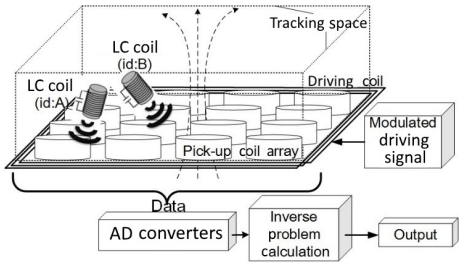


Figure 3: *Tracking principle of IM3D*

4. Implement Initializer for IM3D

In this section, we introduce the implementation of our initializer for IM3D as an application example of our method.

4.1. IM3D

IM3D is a unique magnetic motion tracking system with multiple tiny, identifiable, wireless, occlusion-free passive markers (i.e., LC coils) that provide reasonable accuracy, a reasonable update rate, and an appropriate working space for natural, dexterous 3D interaction.

As shown in Fig. 3, once a varying electromagnetic field is generated by the modulated current in the driving coil, the LC coil inside the electromagnetic field is induced and generates a resonant magnetic flux. Then the pick-up coils in several different locations sense the magnetic field from the LC coil. The signals are measured by AD converters, and finally the program computes the LC coil's 3D position and orientation except for the rotation around the axis of the LC coil. Each LC coil is identifiable since it has an individual number of wire turns and generates magnetic flux in a unique resonant frequency. 15 identical markers are concurrently available in IM3D because the number of unique frequencies of modulated current is limited by the modulator specifications.

This system computes the spatial position and orientation of the LC coils by solving an inverse problem. Assuming that the flux density generated by the marker can be regarded as a magnetic dipole field, more than six values (in an actual implementation, the number of values equals the pick-up coils) of flux density at the known positions are required to calculate the six parameters of the markers: position (x, y, z), orientation (θ and ϕ), and the magnetic moment (M). The system solves this inverse problem, using the following equations (derived from the Biot-Savart law) and a non-linear method of least squares with the affect optimization of the Gauss-Newton method [NK82]. The details of this process were previously described [HTY*08]:

$$S(\vec{p}) = \sum_{i=1}^{n} \left| \vec{B}_{meas}^{(i)} - \vec{B}_{cal}^{(i)}(\vec{p}) \right|^2 \rightarrow Minimum \tag{4}$$

$$\vec{B}_{cal}^{(i)}(\vec{p}) = \frac{1}{4\pi\mu_0} \left\{ -\frac{\vec{M}}{r_i^3} + \frac{3(\vec{M}\cdot\vec{r}_i)\cdot\vec{r}_i}{r_i^5} \right\} \tag{5}$$

$$\vec{p} = (x, y, z, \theta, \phi, M) \tag{6}$$

As the Biot-Savart law indicates, if the LC coil is perpendicular to the magnetic field of the driving coil (i.e., parallel to the pick-up coil array in this case), then it fails to be driven and will not generate the resonant magnetic field: hence tracking loss (dead-angle problem). This problem was successfully solved [HMT*15] by designing a marker with three LC coils in different spatial rotations so that for any time instance, there is at least one tracked LC coil (details are available [HMT*16]). However, this solution suffers when the markers become bulky and the number of available markers decreases. To avoid these problems, a single LC coil must be used as a marker.

Although this system uses a previous result as an initial value, tracking gets lost once the marker goes outside the tracking space, and the previous result becomes improper as an initial value, and thus a robust initializer is required.

4.2. Data Structure

The input data for random-forest consist of 32 voltage values from 32 pick-up coils of the system. Due to the initializer's purpose, the output has the same form as the initial value for the inverse problem (described in section 4.1).

Even with the same spatial pose, the flux strength generated by each LC coil is different due to the different number of wire turns and the various magnitudes of the power supply from the driving coil. We measured the data from different LC coils and found that the normalized distribution of the flux is very similar when the markers have identical spatial configuration. This idea can also be explained by the Biot-Savart law; the magnetic moment of the source determines the scale of the distribution. Therefore the data set of each sample is normalized and passed into the random-forest, helping the initializer discard the massive input samples due to the wide range of the possible magnetic moment.

4.3. Sampling and Training

IM3D's tracking principle is technically a process that searches for a spatial pose where a measured value set matches the theoretical value set, which is computed with the Biot-Savart law. In ideal cases, these two value sets perfectly match. For practical cases, they are also very close because the square error is usually less than 0.001 (position in meters, and rotation in radius). Based on this result, we computed the theoretical values of all the locations and orientations and used them as training samples. We built a generator that enumerates all the spatial poses with high resolution for the whole tracking space and created 32 values in 32 locations of the pick-up coils based on the Biot-Savart law. With this generator, a billion different samples can be created with very little effort for acceptable accuracy. For other 3D motion tracking systems, as far as a known theoretical model exists, training samples can be generated that consist of the position and the orientation of the marker and sensor values.

4.4. Initializer Implementation

We implemented a random-forest model as a classifier whose output is discrete. Since we only require an approximation from the

initializer, this model still satisfies our need. We therefore specifically constructed and configured the forest training parameters for this goal.

We simply constructed a forest for each dimension of the initial value (thus 6). For each positional (x, y, and z), rotational (θ, ϕ), and magnetic moment (M) dimension forest, we tagged the training samples based on target resolution *res*. For instance, tag *ID* of sample A in forest of x can be obtained by applying the following formula (*min* and *max* show the minimum and maximum values of tracking space in the dimension):

$$ID_A = \frac{x - min}{res} \tag{7}$$

In this way when we get predicted tag ID_X from the initializer, we can transform it back into an actual number in a reversed way:

$$x = ID_A \cdot res + min \tag{8}$$

Therefore, the number of classes for each forest (*number*) can be obtained:

$$number = \frac{max - min}{res} \tag{9}$$

Based on our observation in pilot study, since the optimization process highly converges with initial values within 70 mm of the real position, we set our initializer's resolution to 10 mm. For the rotation, we set the resolution to 10 degrees to get as many samples as possible with acceptable accuracy.

We chose the number of trees in each forest based on the number of classes. We experimentally decided the number of trees as half of the number of classes. The other parameter, maximum depth, was also chosen from our experiment results. A detailed comparison is discussed in section 6.3.

5. Evaluation

In this section we describe the evaluation of our method that was implemented in IM3D. Through these evaluations we show the benefit of our initializer.

5.1. Convergence

Convergence is the most important feature brought to the system by our initializer. Previously without an initializer, when tracking loss occurs, the system has trouble recovering since it cannot find a proper initial value for the optimization process. With our method, the system can ensure a proper initial value, and so the optimization process highly converges when the S/N ratio is acceptable. We experimentally proved this feature and the superiority of our method by comparing our method and two other settings: 1) Always setting the initial value to a static position in the tracking space (x = 0 mm, y = 50 mm, z = 0 mm from the IM3D's origin), and 2) randomly choosing a value inside the tracking space at every time instance. We chose these two because they are the most common methods used by practical real-time motion tracking systems when tracking loss occurs.

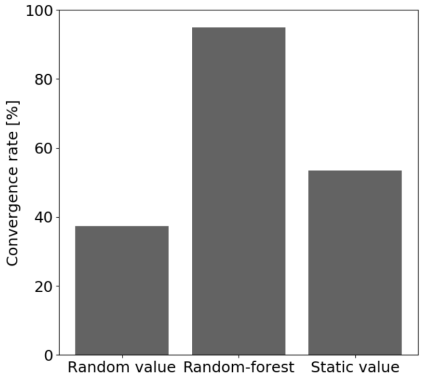


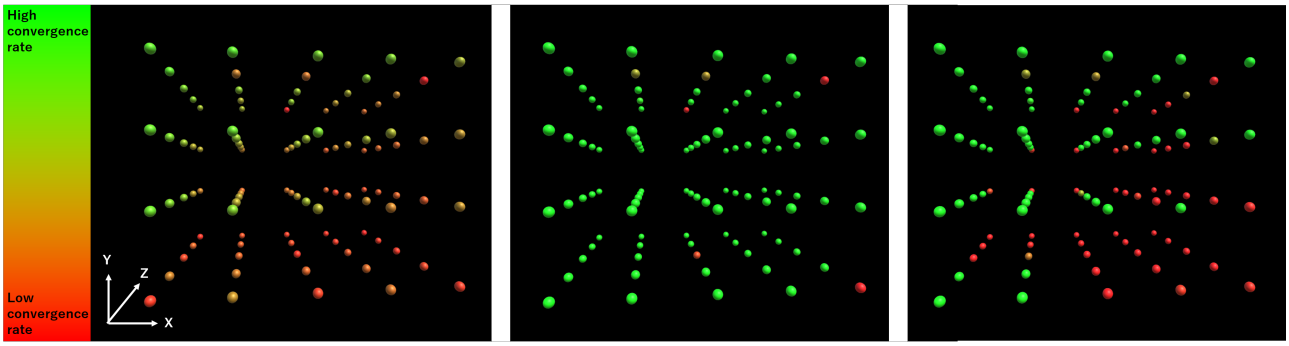
Figure 4: *Convergence rate of three methods*

We put a marker at 100 different locations inside the tracking space to uniformly collect data (3 mm, 32 mm, 64 mm, 93 mm, and 122 mm for x and z axis; 32 mm, 64 mm, 93 mm, and 122 mm for y axis). The real position was measured by a ruler to maintain 1-mm accuracy. The locations are within one quarter of the whole space because of the symmetry of the pick-up coil layout and evenly distributed inside the space while maintaining differences from typical locations and training samples. We mainly evaluated in several height layers from 32 mm (the lowest height our measurement structure can reach) to 122 mm. We measured flux 100 times in each position and processed the optimization with initial values computed by the three methods. Diverged processes were detected by checking whether the calculated magnetic moment of each process was significantly large. To avoid counting trials in which the processes converged into the local minima of the cost function, we also counted trials as diverged trials when the distance between the calculated position and the actual position was larger than 40 mm.

The convergence rate ($\frac{trials - diverged}{trials}$) of each method is shown in Fig. 4, and the convergence rate of our method significantly exceeds all other methods. Individual results in every specific position are shown in Fig. 5. Our initializer has more effective results in almost all the measuring positions, indicating that our initializer can provide better initial values even for areas far from the center of the space. Hence it is more robust. This result also implies that our method will be more effective for other tracking systems with a larger tracking space. Therefore, the initial values predicted by our method are perspective values for computation.

5.2. Prediction

The system with our initializer has same tracking accuracy as a previous work [HMT*15] that applied the same principle. However, our random-forest method shows amazing potential in position calculation since the initializer's output seems very close to the final result. Additionally, we want to confirm that the random-forest yields a prediction that is close to the actual location, so that the inverse problem can be successfully solved. Based on these goals we experimentally demonstrated this feature.



(a) Random value (b) Random-forest (c) Static value

Figure 5: *Visualization of convergence rate of three methods*

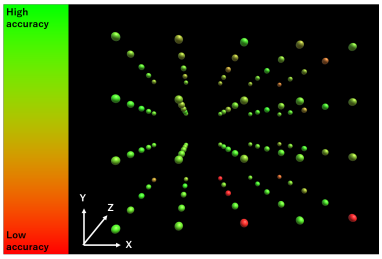


Figure 6: *Visualization of prediction accuracy inside space*

5.2.1. Positional Accuracy

We calculated the distances between the initial values predicted by random-forest and the actual positions by using the data we acquired in section 5.1.

The visualized results are shown in Fig. 6. The error of each location is mapped in 3D colored dots, and the minimum error (30 mm) is shown as green dots and the maximum error (150 mm) is red dots. Most points are either highly green or highly red. The average prediction error within the tracking space is 35 mm, which is highly satisfactory as an initial value for the inverse problem, since from previous experience an initial value with error less than 70 mm can ensure the convergence in calculation with an acceptable S/N ratio.

5.2.2. Flux Strength Affect

We conducted an additional evaluation with different rotations and different numbers of markers. These different conditions changed the flux sensed by the pick-up coil. We want to evaluate the magnitude of flux with which the initializer can yield a reliable result.

We used a rotating platform (Fig. 7) to fix the marker in corresponding rotations. In six different locations in the tracking space, we got the data with both one-marker and fifteen-marker configurations. We defined 90 degrees as a parallel pose to the plane of the pick-up coil array and 0 degrees as a perpendicular pose to the plane. The magnitude of flux decreases when the marker's pose becomes close to 90 degrees, and the S/N ratio becomes poor.

Fig. 8 shows the prediction error in different rotation from various locations, where the error in different locations (coordination represented as X|Y|Z at the top of the graphs) is shown as lines in

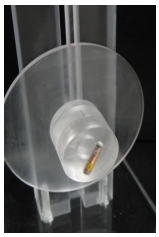
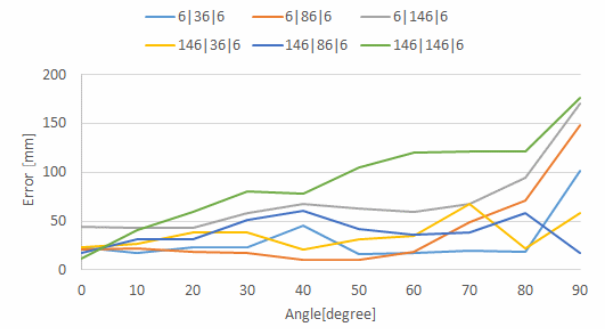
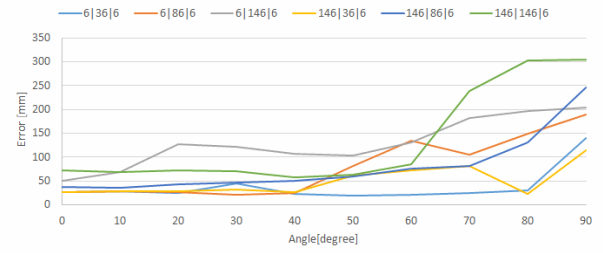


Figure 7: *Marker (LC coil) on rotating platform*

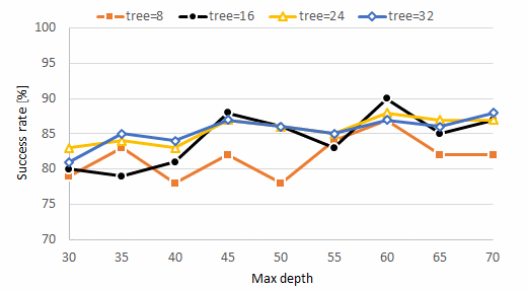


(a) one-marker configuration result

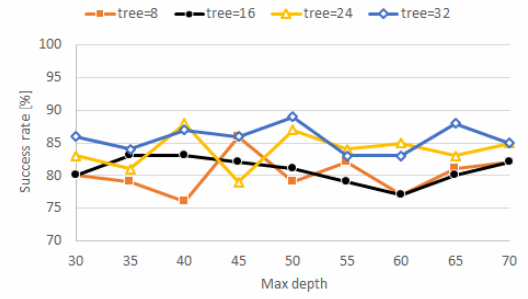


(b) fifteen-marker configuration result

Figure 8: *Prediction error in different rotations from different locations*



(a) forests with 10 mm interval training samples



(b) forests with 5 mm interval training samples

Figure 9: *Success rate with meta-parameters (i.e., maximum depth of trees and number of trees)*

different colors. These results show that prediction accuracy falls when the flux is reduced, since the accuracy in the larger rotation is lower than that in the smaller ones, and in the same pose, the accuracy in the one-marker configuration is higher than the fifteen-marker configuration. Actually, the IM3D system suffers from this dead-angle problem. When its angle is almost parallel to the pick-up coil array, the LC coil cannot generate any flux, which also caused tracking loss even with proper initial values.

5.3. Meta-Parameters of Random-Forest

We also did another experiment to obtain output from our initializer with the measured flux data in experiment in section 5.1 to determine the effects of different parameter configurations. We focused on the estimation success rate, which is defined as the prediction percentage with error smaller than 50 mm among all the test points. Fig. 9(a) shows the success rate when we train the forests with 10-mm interval samples (1.1 million samples) with a maximum tree depth (30 to 70) for four different number of trees (8, 16, 24, 32). This graph shows the results with different combinations of number of trees and maximum depths for each tree. For all the configurations of different numbers of trees, the success rate reached its highest result with a maximum depth of 60, and with 16 trees the forest reaches its highest rate of 90%.

Fig. 9(b) shows the same experiment with 5-mm interval samples (8.8 million samples). Similar to Fig. 9(a), it has a peak for the success rate, and more trees increase the success rate.

We also found that the accuracy with 10-mm samples is better

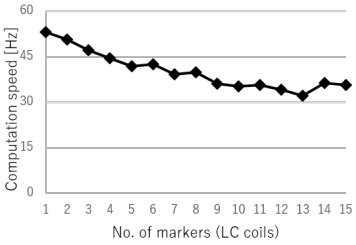


Figure 10: *Speed (in FPS) decreases when markers (LC coils) increase*

than 5 mm by comparing all these results, probably caused by more ambiguity in the excessively massive data samples.

5.4. Performance

For a real-time motion tracking system, since its computational speed's performance must be ensured, we also did such experiments. We mainly tested the speed of the entire process (our initializer and the optimization process) and the initializer's speed itself.

For the random-forest performance, we simply ran the random-forest evaluation 1000 times and checked the time cost. Each prediction call cost 1 millisecond, which barely affected the entire system's performance.

For the speed of the whole process, we set up the system with a different number of markers ranging from 1 to 15, ran it for 100 seconds, and tracked all the used markers. The system provided high-tracking speed close to 60 Hz for a marker. When tracking with up to 15 different markers, it can still maintain a speed over 30 Hz (Fig. 10). The speed reduction is caused by the increase of overhead, other than introduced by our initializer.

6. General Discussion

Due to resource limitations, we only implemented our method on a magnetic 3D motion tracking system. However, as mentioned in section 3, it can generally be applied to other optimization-based tracking systems. As in these systems, since simulation is easier than solving the inverse problem, massive training samples can be easily obtained from the simulation. For example, for camera-based optical tracking systems, the simulation input can be rendered as images with specific camera-marker configuration, and for magnetic tracking systems, the simulation input can be theoretical values with specific marker locations and rotations.

Our evaluation results proved that our method adds robustness to 3D motion tracking systems. Since interactive techniques always require continuous tracking results for continuous interaction, our method, although not directly, will also improve the experience of 3D motion-based interactive techniques.

We chose to experiment with a random-forest rather than other data-driven methods, such as a deep neural network (DNN) or the Gaussian process. Actually, we tested these methods with the same input-output strategy and training data through preliminary research. However, none of these methods yielded satisfactory results, perhaps due to the complexity and the ambiguity of such inverse problems or the model's complexity during run-time. On the

other hand, without yielding very accurate results, random-forest constantly gave acceptable predictions very quickly; we choose it because it is a fast and accurate initializer.

Regarding the run-time phase, random-forest works like lookup tables, which only search from existing data to find the best match. Since we can generate a very large database using the Biot-Savart law without much effort, such a method might effectively get close output. Based on this difference, perhaps other similar methods (for instance, KD trees) might be successful, although further experiments are needed.

Our evaluation shows that our initializer itself is so fast that there is almost no effect on the system's cost. This leaves space for further improvements, such as filters or regression.

7. Conclusion and Future Work

We present a novel random-forest-based initializer for optimization-based 3D motion tracking systems. As an example, we apply this approach to a magnetic 3D motion tracking system (IM3D) to solve an actual initialization problem. Our fast and accurate initializer provides successful per-frame initialization from 32 sensor values (flux intensity) for an optimization problem in real-time. With this feature, the new system gains recovery ability, allowing it to recover from tracking loss despite very low computation resource cost.

We propose and discuss the structure and the parameter configuration of our initializer and experimentally evaluate its performance. It gives an initial value within 35 mm from the actual position (in most cases in 1 millisecond), which helps the optimization process calculate a result with less than 10 mm error. Since our new initializer can yield output within 1 millisecond, the integrated system's speed exceeds 30 Hz, even with 15 markers, which is reasonable for real-time motion capture or applications.

The accuracy of the output from our initializer and many other data-driven tracking systems convinces us that perhaps we can completely rely on data-driven methods instead of such optimization problems of this system. This could boost this system's speed, but it requires more strategy design and training efforts. A possible solution is to divide the whole tracking space into many small regions to build a multi-layer predictor. Other machine-learning researches have proven the efficiency of this idea for increasing accuracy.

With our initializer, real data can also be collected for on-line training. For future applications, we are considering acquiring samples during run-time for progressively refining the model. Moreover, dexterous motion data can also be collected for other researches such as motion synthesis.

Other future work will apply our approach to more different tracking systems such as optical and acoustic systems.

8. Acknowledgments

This work was supported in part by JSPS KAKENHI Grant Number 15H01697.

References

[Bis06] BISHOP C. M.: *Pattern recognition and machine learning*. springer, 2006. 1

[CV95] CORTES C., VAPNIK V.: Support vector machine. *Machine learning 20*, 3 (1995), 273–297. 2

[HMT*15] HUANG J., MORI T., TAKASHIMA K., HASHI S., KITAMURA Y.: IM6D: Magnetic Tracking System with 6-DOF Passive Markers for Dexterous 3D Interaction and Motion. *ACM Trans. Graph. 34*, 6 (Oct. 2015), 217:1–217:10. 2, 4, 5

[HMT*16] HUANG J., MORI T., TAKASHIMA K., HASHI S., KITAMURA Y.: 6-DOF Computation and Marker Design for Magnetic 3D Dexterous Motion-tracking System. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology* (New York, NY, USA, 2016), VRST '16, ACM, pp. 211–217. 4

[HTHK14] HUANG J., TAKASHIMA K., HASHI S., KITAMURA Y.: IM3D: Magnetic Motion Tracking System for Dexterous 3D Interactions. In *ACM SIGGRAPH 2014 Emerging Technologies* (New York, NY, USA, 2014), ACM, pp. 12:1–12:1. 2

[HTY*08] HASHI S., TOYODA M., YABUKAMI S., ISHIYAMA K., OKAZAKI Y., ARAI K. I., KANETAKA H.: Wireless magnetic motion capture system using multiple LC resonant magnetic markers with high accuracy. *Sensors and Actuators A: Physical 142*, 2 (2008), 520–527. 4

[KSH12] KRIZHEVSKY A., SUTSKEVER I., HINTON G. E.: Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105. 2

[MFWM17] MCINTOSH J., FRASER M., WORGAN P., MARZO A.: Deskwave: Desktop interactions using low-cost microwave doppler arrays. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, 2017), CHI EA '17, ACM, pp. 1885–1892. 2

[NK82] NAKAGAWA T., KOYANAGI Y.: *Experimental Data Analysis by the Least Square Method*. The University of Tokyo Press, 1982. 4

[PK07] PINTARIC T., KAUFMANN H.: Affordable infrared-optical pose-tracking for virtual and augmented reality. In *Proceedings of IEEE VR Workshop on Trends and Issues in Tracking for Virtual Environments* (2007). 1, 2

[Pol] POLHEMUS: http://polhemus.com/. 2

[RBSJ79] RAAB F., BLOOD E., STEINER T., JONES H.: Magnetic position and orientation tracking system. *IEEE Transactions on Aerospace and Electronic Systems AES-15*, 5 (Sept. 1979), 709–718. 1, 2

[SFC*11] SHOTTON J., FITZGIBBON A., COOK M., SHARP T., FINOCCHIO M., MOORE R., KIPMAN A., BLAKE A.: Real-time human pose recognition in parts from single depth images. In *Proceedings of IEEE Computer Vision and Pattern Recognition (CVPR)* (2011), pp. 1297–1304. 2

[SKR*15] SHARP T., KESKIN C., ROBERTSON D., TAYLOR J., SHOTTON J., KIM D., RHEMANN C., LEICHTER I., VINNIKOV A., WEI Y., FREEDMAN D., KOHLI P., KRUPKA E., FITZGIBBON A., IZADI S.: Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (New York, NY, USA, 2015), CHI '15, ACM, pp. 3633–3642. 2

[THF*03] TONG W., HONG H., FANG H., XIE Q., PERKINS R.: Decision forest: combining the predictions of multiple independent decision tree models. *Journal of Chemical Information and Computer Sciences 43*, 2 (2003), 525–531. 2, 3

[ZCX12] ZHAO W., CHAI J., XU Y.-Q.: Combining marker-based mocap and RGB-D camera for acquiring high-fidelity hand motion data. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar Germany, Germany, 2012), SCA '12, Eurographics Association, pp. 33–42. 2