SWITCHING ATTENTION LEARNING – A PARADIGM FOR INTROSPECTION AND INCREMENTAL LEARNING

D. Zhang[1], *B. Nebel*[2], *A. Hornung*[3]

[123]University of Freiburg
Georges-Khler-Allee, Geb. 52
Freiburg im Breisgau, 79110, Germany

ABSTRACT

Humans improve their sport skills by eliminating one recognizable weakness at a time. Inspired by this observation, we define a learning paradigm in which different learners can be plugged together. An extra attention model is in charge of iterating over them and chosing which one to improve next. The paradigm is named Switching Attention Learning (SAL). The essential idea is that improving one model in the system generates more "improvement space" for the others. Using SAL, an application for tracking the game ball in a table soccer game-recorder is implemented. We developed several models and learners which work together in the SAL framework, producing satisfying results in the experiments. The related problems, advantages, and perspective of the switching attention learning are discussed in this paper.

1. INTRODUCTION

Attention plays a very important role during the learning processes of human beings. For example, students in a university take several courses in a morning. The subjects could be dependent or independent. The understanding of physics requires some background in mathematics. Studying English is not directly related to learning biology. In any situation, the students can develop their knowledge by switching their attention from one subject to another. Switching attention can be observed not only in complex tasks as in the example above but also in some much simpler ones such as learning a one-second sport action. For instance, a human player tries to improve his dart for six months. M. Suwa found that together with a measurable improvement of his overall performance, the player switched his attention to different parts of his body, e.g. the waist, the elbow, or the fingers. Based on the "switched attention", the key mechanism of the improvement is explained as the so-called "Meta-Cognitive Verbalization" [1].

Humans have the ability of learning to improve their skills. When facing a learning task, such as playing darts, people can acquire the necessary skills by repeating the following three steps: finding a weakness, overcoming it, and switching attention to another weakness. Iterating over the steps can be regarded as an incremental-learning process with introspection. In artificial intelligence (AI), a learning process can normally be described as acquiring a model and its parameters. There exist many learning approaches which have been proved to be very useful and powerful. However, to our knowledge, there exists no approach which has a switching attention mechanism similar to what humans have. And no agent has learning abilities comparable to the introspection and incremental learning of human beings. This work is motivated by these observations.

In the **Switching Attention Learning (SAL)** paradigm, there are a series of learners, each of which can be implemented by any existing learning approach. They are cooperative for a task by using a switching attention mechanism. SAL mimics the attention-related structures in human intelligence.

In this paper, first, we define the SAL paradigm in Chapter 2. The essential idea is to make different learners interact with each other. One learner can produce more "improvement space" for the others, and this improvement can be iterated for the incremental learning. Then, we describe the first application by using the idea, which is explained in detail in Chapter 3. The task is to track the game ball in a table soccer game recorder. Finally, we draw the conclusion in Chapter 4.

2. SWITCHING ATTENTION LEARNING

A *switching attention learning* (SAL) system is defined by the following **four elements**:

- a set of system goals, $\mathbf{G}$;
- a set of models, $\mathbf{M}$, cooperating for $\mathbf{G}$;
- a set of data and structures $\mathbf{D}$, being input and output of the models in $\mathbf{M}$;
- a set of learners $\mathbf{L}$, which can improve the models in $\mathbf{M}$ towards G.

A model m is an **active model** if there is a learner l_m improving m, where $m \in \mathbf{M}$, $l_m \in \mathbf{L}$. The input of the model "m" is denoted as I_m and its output as O_m, where $I_m, O_m \subset \mathbf{D}$. So far, an active model can be defined by a

4-tuple, (m, l_m, I_m, O_m). If the set of the active models is $\mathbf{M_a}$, we have $\mathbf{M_a} \subseteq \mathbf{M}$.

We use "d" to denote an element in $\mathbf{D}$. d is an **active element** if $d \in I_{m_1}$ and $d \in O_{m_2}$ where $m_1, m_2 \in \mathbf{M_a}$ and $m_1 \neq m_2$. With "$\mathbf{D_a}$" denoting the set of all active elements, d_i and d_j are "**connected**" by m if $d_i \in I_m$ and $d_j \in O_m$, where $d_i, d_j \in \mathbf{D_a}$, $m \in \mathbf{M_a}$. There is a **path** from d_1 to d_n if $\forall d_k, d_{k+1} \in \{d_1, d_2, ..., d_n\}$, d_k and d_{k+1} are connected. A system complies with the **switching attention learning** paradigm, if there exists a path from an active element to itself.

The definition of SAL limits our domain to closed-loop multiple learner systems. *Closed-loop* is a concept which is widely used in control theory and artifical neural networks. The idea is to notify a system with feedback of its own performance. *Closed-loop* is important for the SAL paradigm because the learning process of human beings, which inspires the initial idea of SAL, appears to be closed-loop. We believe it will gain advantages for the topics in incremental learning.

We can distinguish a SAL system from others by its definition. For instance, in the Boosting approach [2], there are two learners. One learner *boosts* a series of *weak classifiers* from the training set in which examples are weighted by their *importance*. The other finds a combination of the learned classifiers for the final classification. Boosting is not SAL because the first learner does not take any input from the second one, although the two learners interact in the way that the combination model uses the output of the weak classifiers as its input. There are similar learning systems in which different learners are involved. For example M. Fox and her colleagues developed an approach [3] where the structure of Hidden Markov Models (HMMs) can be learned by a learner using Kohonen networks. The parameters of the HMMs are computed by another learner using Expectation Maximization framework. These approaches are different from ours because the "active elements" in the system do not form a loop.

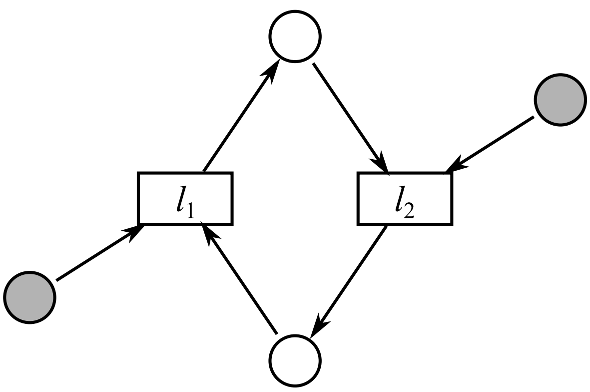


Figure 1: An Example with Two Learners

SAL requires at least two learners. Figure 1 shows an example. The rectangles are learners, the circles are their inputs and outputs. The white nodes are the *active elements*. They can be regarded as the medium of the improvement. If we ignore the two grey nodes and their peripheral arrow lines, the remaining nodes and lines, which form a loop, show an example of the smallest possible SAL.

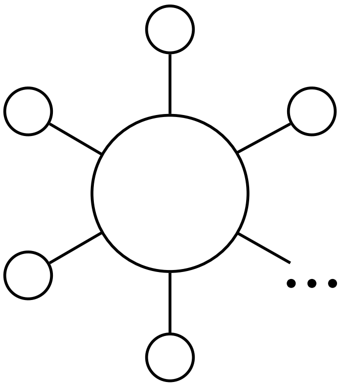


Figure 2: Active Elements

SAL provides a flexible context in which a specific real-world problem can be divided into sub-problems, each of which can be solved in an individual model. The development process can pass three stages. First, the sub-problems are solved independently, with the models being defined with their input and output. In this stage, we can expect an open-loop system, in which the models are independent, or they can do only a few interactions but not incrementally. Then, we focus on the elements which can bridge different models, reforming them so that they can represent both input and output. Figure 2 shows the development map of this stage. The big circle at the center can be regarded as the closed-loop learning. We need to find out the *active elements* in the system and develop learners for them. Finally, the dependency among the models and learners are analyzed. We use a dependency map to show these relations. Figure 3 shows the dependency map of the two-learner example. We can design an attention model according to the topology of the dependency map. In the example, the learners should be simply performed one after another. We believe designing a good attention model can accelerate the learning process in a more complex context.

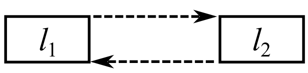


Figure 3: Dependency Map

SAL provides a framework for exploring the relations among different models and learners. The essential characteristic of SAL is that the structure based on the models and learners produces the power of introspection and incremental learning. SAL is supposed to be used in scenarios where achieving final goals requires a few intelligent components, each using distinct representation and algorithms, solving problems from different points of view. It also provides a platform where different solutions for the same problem can be compared and combined. One of the possible applications is the table soccer robot [4]. The agent should not only play against humans, being adaptive to different human players, but also classify them and learn from them. SAL can also be used in much smaller applications. Such an application is included in Chapter 3.

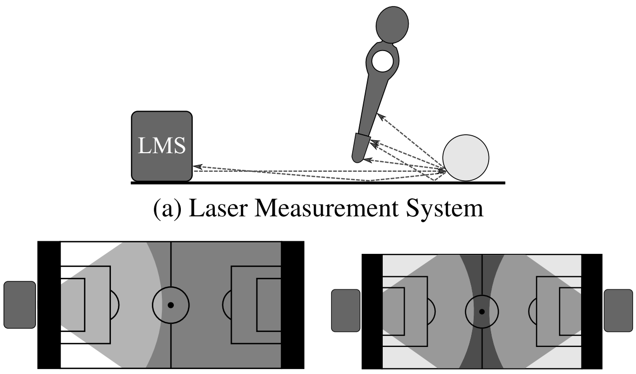
So far, we defined the basic concepts of SAL, showed some possible tools during the developing process, and illustrated some application domains. However, there are many problems remaining. One problem is that improving a single learner in the system does not guarantee a better performance of the whole system. Another problem is that the errors generated by each learner are possibly accumulated in the system, causing mistakes later. Although we expect that the difficulties should be overcome by designing an evaluation model towards avoiding them, there are no experiments or solution yet. In addition, developing many models and structures normally requires a long period of time, which makes it difficult to try or prove the concepts in SAL. Nevertheless, we believe the problems will be met and can be solved on-the-fly.

3. TRACKING A SINGLE TARGET USING SAL

Tracking problems exist in many domains thus there are many well-developed approaches. The Kalman Filter (KF) framework is one of the most successful and effective methods [6]. It can be extended to different forms and integrated with other methods. Interacting multiple model (IMM) contains a set of KFs as well as Markov Decision Processes (MDPs) for tracking one or more targets which have several *moving modes* [7]. In *switching Kalman filters*, a switch node is introduced into the topology network of the KFs to model the dynamic changes of the movement [8]. These methods can also be applied for our ball tracking problems. But they are not as flexible as SAL when domain knowledge is taken into consideration. There is another category of approaches based on particle filters. These approaches are proved to be very powerful in mobile robots [9], but they require remarkable computational power, which is not suitable for our application. In this section, we describe the method for tracking the ball of table soccer to illustrate how SAL is applied in a real application.

3.1. The Tracking Problem

KiRe is a table soccer game recorder. It provides a way to record human-played games[5]. In KiRe, the position of the ball is measured by a **laser measurement systems** (SICK **LMS**-400), as illustrated in Figure 4(a). There is a gap of about $12mm$ between a playing figure and the game field. We installed the LMS behind the goal of the game table. The laser beam goes through the gap, and targets the lower part of the ball. The receiver at the upper part of the *opening* on the LMS receives the reflected signals. The opening angle of the laser beam is $70\,^\circ$, and the valid measurement range of the LMS is from $700mm$ to $3000mm$. Each data slice contains 280 measurements of distance and angle evenly distributed over the open angle. Figure 4(b) shows the bird-view of the game field. With one LMS, the field is divided into three different regions. Two corners in the left side are outside the laser's view. The measurements are invalid in the grey fan region. The remaining dark grey region is within the valid range of the LMS, covering the right half of the field. By removing the background from the laser view, we can compute the ball's position.



(a) Laser Measurement System

(b) By One LMS (c) By Two LMSs

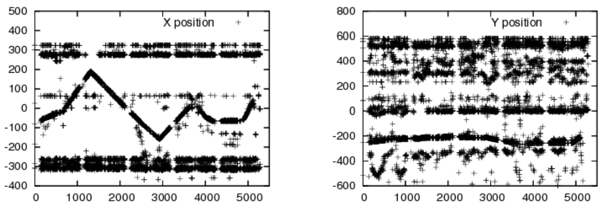
Figure 4: Measuring the Position of the Game Ball

In order to measure the whole field, we mounted two LMSs symmetrically in KiRe. These two LMSs communicate via Ethernet, which synchronizes their laser scans. Figure 4(c) shows the different regions in this situation. In Table 1, we list the types of the regions and the information for locating the ball. When the ball is in the grey regions, one of the LMSs gets the invalid distances at the laser spots on the ball. However, the angle of these invalid data still provide information about the ball's position. Therefore, we can still fuse the measurements from two LMSs to compute the intended position.

Color	Description
Dark Grey	Both LMSs have valid measurements
Grey	One LMS is valid, the other is invalid
White	One LMS is valid, the region is out of the other's view.

Table 1: Regions and Their Information

The LMSs scan the field with a frequency of 350 Hz. With two LMSs synchronized, the computer needs to process 700 data frames per second. We recorded a segment of data of about 5 seconds during which a human player dribbles and kicks the ball. Figure 5 shows the recorded data. The horizontal axis is the time in millionseconds, the vertical axis is the position in millionmeters. The x and y positions are shown in separate plots. There are several possible positions in a single data slice because of the noise. Thus we got more than 7000 points all together.



(a) X Positions (b) Y Positions

Figure 5: Raw Data from the Sensor

The raw data is very noisy, which makes the tracking task very challenging. Below, we list the main difficulties.

- When two LMSs are face to face, they interfere with each other even if they are synchronized, because the laser beam of one LMS is reflected by the mirror of another.
- The game figures, which are moved and turned constantly, disturb the reception of the laser signals.
- The ball jumps frequently, and cannot be observed by the LMSs when doing so.
- We do not use a real-time operating system, thus there are some processing gaps of about $80ms$ in the data.

3.2. The Implementation

We developed a **ball module** using SAL for the tracking task, as shown in Figure 6. There are three models in the module. A **sensor model** is constructed for filtering the noise. A point in a data slice is classified as valid or noise. The noise will be discarded. Each data slice contains one valid point at most. A **segment model** is developed for segmenting the data sequence into much smaller parts. In each segment, the ball is assumed to have constant acceleration and movement direction. A set of **Kalman filters** are implemented to smooth the data within a segment. The raw data is processed first by the *sensor model*. The sequence of the valid points are forward to the *segment model*. When a new segment is detected, the *Kalman filters* are *reset* so that they can adjust to the sudden changes in the ball's movement.

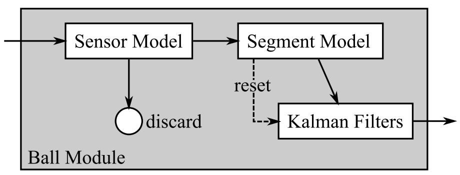


Figure 6: The Ball Model

3.2.1. Sensor model

An approach based on a decision tree is implemented in the sensor model for the classification. Clusters are constructed according to the dynamic updates of the data sequence. In each cluster, the data points are close to each other. With the help of the clusters, the decision tree can be constructed based on boolean attributes which can be obtained by answering the listed questions.

- Is the data point supported by both LMSs? The point is called a **full-belief point** in the true case.
- Does the data point belong to a cluster which has the maximum point number among all the clusters?
- Does the data point belong to a cluster which contains full-belief points?
- Is the point in a region where the noise points are detected with high probability? The four corners are this kind of region because each corner can be observed by only one LMS.
- Does the point belong to a cluster which is updated very often?
- Does the point belong to a newly-created cluster?
- Will the point generate high innovation if it is used to update the Kalman filters?

3.2.2. Kalman filters and segmentation

The data from LMSs is a distance-angle pair. We need to transform them to (x, y) position. Because of the transformation, extended KFs should be employed for the tracking. However, we choose discrete KFs. The reasons are discrete KFs save computational power significantly, when they are used separately for x and y. And they do not perform worse than an extended KFs in the experiments. The discrete KFs are updated according to the time and measurement update functions which are already standard [6]. We skip them here.

By considering the position, velocity and acceleration, we implemented an approach using *triple-integrator* KFs which is widely used and has excellent performance in many applications [10]. Here we only explain it briefly. Equation 1 defines the vector of x, where v_x is the velocity along x direction, a_x is the acceleration. The update of $\vec{X}$ is governed by Equation 2, with A_x is defined in Equation 3. Δt is the time span since last data slice. The process noise w_k is defined in Equation 4, where the Q matrix can be computed dynamically by Equation 5. The belief factor b can be adjusted to trade-off the prediction and the measurement.

$$\vec{X} = \begin{pmatrix} x & v_x & a_x \end{pmatrix}^T \quad (1)$$

$$\vec{X}_k = A_x \vec{X}_{k-1} + w_{k-1} \quad (2)$$

$$A_x = \begin{pmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{pmatrix} \quad (3)$$

$$p(w) \sim N(0, Q) \quad (4)$$

$$Q = b \cdot \begin{pmatrix} \frac{1}{20}\Delta t^5 & \frac{1}{8}\Delta t^4 & \frac{1}{6}\Delta t^3 \\ \frac{1}{8}\Delta t^4 & \frac{1}{3}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{6}\Delta t^3 & \frac{1}{2}\Delta t^2 & \Delta t \end{pmatrix} \quad (5)$$

3.2.3. The evaluation model

The ball module has the functionality of classifying, segmenting, and smoothing the data sequence. These functionalities are based on the data from the past. From another point of view, we can evaluate the performance of the ball module without any temporal limitation. For example, both the data from the past and the data from the future can be used for classifying the current point. According to this principle, we implement an evaluation model which follows three rules listed below.

- A valid data segment should contain at least three full-belief points.
- A point belongs to a data segment if its distance to the closest point in the segment is within $3mm$.
- The more points are included, the better the performance the system gets.

3.2.4. The active elements and the learners

With the help of the evaluation model, we can develop several learners to improve the performance of the system. As the medium for the improvement, four *active elements* are found. They are shown in Figure 7. **decision tree** is used for the classification in the sensor model. **segment thresholds** are the time and innovation thresholds in the segment model. **Belief factor** trades off the prediction and measurements in triple-integrator KFs. This factor affects the innovation threshold used in the segment model, which will reset KFs when a data point generates a high-grade innovation. **Training set** can be regarded as the results of the evaluation. Each example in the set contains seven attributes and a class label. The attributes are provided by the sensor model. The class label comes from the evaluation. Four learners are implemented to learn the active elements from the data. The learning tasks are listed as below.

1. Learning the training set from the output of the ball module
2. Learning the decision tree from the training set
3. Learning the segmentation thresholds by maximizing the point number in the output of the ball module.

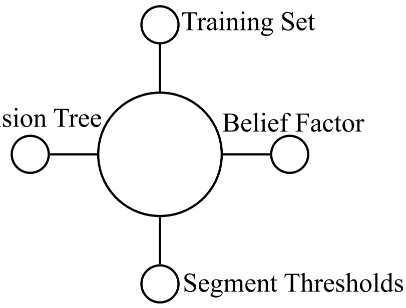


Figure 7: Active Elements

4. Learning the belief factor by minimizing the average innovation of the KFs

As there are not so many learners, we developed a simple attention model in which each iteration has seven steps: E, 1, 2, E, 3, E, 4. In an **E** step, the performance of the system is evaluated.

3.3. Experiments

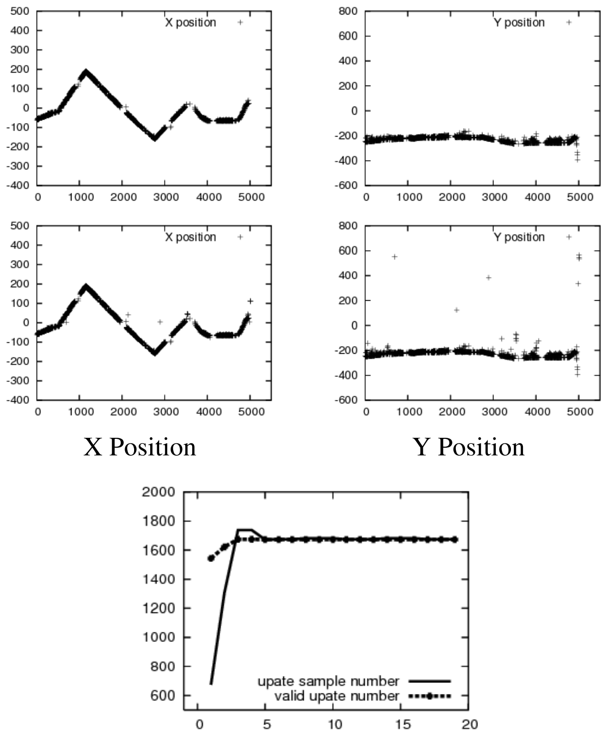
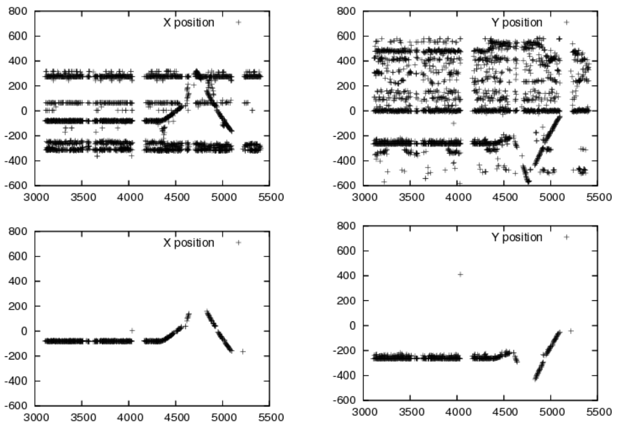


Figure 8: Offline Learning

We use the recorded data shown in Figure 5 for offline learning. The implemented SAL system is run on the data for six iterations. The results are shown in Figure 8. The first row is the valid trajectory of the ball which is obtained manually. The second row shows the output of the learned system. Although there are a few noise-points left in the trajectory, the results are satisfying. This noise remains because the decision tree does not have enough attributes to classify them apart from the valid points. The third row shows the improvements over the iterations. In this figure, the horizontal is the number of the *E* steps that are performed. The solid curve shows the changes of the output number of the ball module. The doted line shows the changes of the valid point number in the evaluation. After about six *E* steps, which are two iterations, the learning converges to the satisfying results. The learning curves indicate that a better ball module enlarges the ability of the evaluation, while a better evaluation improves the ball module.



X Position y Position

Figure 9: The Selected Data Segment in Online Testing

The trained system is tested in an online manner. We found that the ball can be tracked in real-time, and its trajectory is smooth enough. We select a segment of data, which is illustrated in the first row of Figure 9. The output of the system is shown in the second row. In the figure, the ball is lost when it is in a corner. The reason is that the corners are regarded as the noisy regions which are not important for the game. Therefore we can conclude that the trained system stays stable on the unknown data.

4. CONCLUSION AND PERSPECTIVE

In this paper, the Switching Attention Learning (SAL) framework is introduced, in which different learners can be plugged together. The learners are cooperative supporting introspection and incremental learning. The basic principle is that improving one model in the system generates more *improvement space* for the others. We discussed the related problems, advantages, and the application domains of the SAL.

The first application of SAL is implemented for tracking the game ball in table soccer. There are four *active elements* in the application, a decision tree for classification, the training set for building the decision tree, the belief factors for KFs, and the thresholds for segmentation. We developed four learners for these elements and implemented an attention model for the iterations over these learners. The experiments show that the application is successful for the tracking task. The intended *improvement space* is generated by the learners in the system.

In the future, we plan to implement more complex SAL systems. The SAL will be applied in applications such as table soccer robots and playing Tetris against human players.

5. REFERENCES

[1] M. Suwa. *A Congnitive Model of Acquiring Embodied Expertise Through Meta-cognitive Verbalization.* In Proceedings of 1st International Symposium on Skill Science (ISSS 2007), pages 80–86, Keio University, Tokyo, Japan, 2007.

[2] Y. Freund and R. E. Schapire *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting* In Journal of Computer and System Sciences, no. 55, 1997.

[3] M. Fox, M. Ghallab, G. Infantes, and D. Long. *Robot Introspection Using Learned Hidden Markov Models* In Artificial Intelligence, 170(2):59–113, 2006.

[4] T. Weigel and B. Nebel *Kiro – An Autonomous Table Soccer Player* In Proceedings of RoboCup Symposium '02, pages 119 – 127, Fukuoka, Japan, 2002.

[5] D. Zhang and B. Nebel. *Recording and Segmenting Table Soccer Games – Initial Results* In Proceedings of the 1st International Symposium on Skill Science 2007 (ISSS'07), pages 61–67, Tokyo, Japan, 2007.

[6] G. Welch and G. Bishop. *An Introduction to the Kalman Filter* Technical Report 95-041, University of North Carolina at Chapel Hill, 1995.

[7] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan *Interacting Multiple Model Methods in Target Tracking: a Survey* In IEEE Transactions on Aerospace and Electronic Systems page: 103-123 Volume: 34, Issue: 1 Jan 1998

[8] K. Murphy. *Switching Kalman Filters* In Technical report, U. C. Berkeley, 1998.

[9] D. Fox, S. Thrun, F. Dellaert, and W. Burgard *Particle Filters for Mobile Robot Localization* In Sequential Monte Carlo Methods in Practice. Springer Verlag, New York, 2000.

[10] P. R. Bélanger, P. Dobrovolny, A. Helmy, and X. Zhang. *Estimation of Angular Velocity and Acceleration from Shaft-Encoder Measurements* In The International Journal of Robotics Research. Vol. 17, No. 11, pp. 1225-1233, November 1998.