HAL
archives-ouvertes.fr

Data-Driven Identification of Group Dynamics for Motion Prediction and Control

Mac Schwager, Dean Anderson, Daniela Rus

▶ **To cite this version:**

Mac Schwager, Dean Anderson, Daniela Rus. Data-Driven Identification of Group Dynamics for Motion Prediction and Control. 6th International Conference on Field and Service Robotics - FSR 2007, Jul 2007, Chamonix, France. inria-00194927

HAL Id: inria-00194927
https://hal.inria.fr/inria-00194927

Submitted on 7 Dec 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Data-Driven Identification of Group Dynamics for Motion Prediction and Control

Mac Schwager[1], Dean Anderson[2], and Daniela Rus[1]

[1] Computer Science and Artificial Intelligence Lab
MIT, Cambridge, MA 02139, USA
schwager@mit.edu, rus@csail.mit.edu
[2] USDA-ARS, Jornada Experimental Range
Las Cruces, NM 88003, USA

Summary. A decentralized model structure for representing groups of coupled dynamic agents is proposed, and the Least Squares method is used for fitting model parameters based on observed position data. The physically motivated, difference equation model combines effects from agent dynamics, interactions between agents, and interactions between each agent and its environment. The technique is implemented to identify a model for a group of three cows using GPS tracking data. The model is shown to capture overall characteristics of the group as well as attributes of individual group members. Applications to surveillance, prediction, and control of various kinds of groups of dynamical agents are suggested.

1 Introduction

We wish to model groups of coupled dynamic agents, such as flocks, swarms, and herds, using measured data from those agents. For example, we would like to be able to use the trajectories of people in a crowd to develop dynamical models that capture the behaviors of the crowd as a whole. This is a prohibitively complicated problem in general, however, we provide a practical solution by restricting our attention to a special model structure. We propose a difference equation model that is decentralized and nonlinear, though it is designed to be linear-in-parameters. We use the Least Squares method to fit model parameters to position data from a group of agents. Such a model may then be used, for example, to predict future states of the group, to determine individual roles of agents within the group (e.g. leaders vs. followers), or, ultimately, to control the group. This could help ranchers to manage overgrazing by livestock, or allow animal behavioral scientists to determine social roles within animal groups. In the case of people, the ability to model group behavior has numerous applications in surveillance, urban planning, and crowd control. Also, this technique can be used to design controllers for groups of engineered agents to mimic the behaviors of natural agents.

The large body of work on designing and analyzing models of flocks, swarms, and similar decentralized dynamical systems ([3, 7] and references therein) does not

incorporate learning from data to tune such models. However, other related learning problems have been investigated. For example, in [1] a system identification technique is used to model global properties of a swarm of robots over time using observed data from the robots. There also has been considerable activity in learning behavioral models of individual natural agents. In [5] and [6], system identification is carried out on switching linear systems to learn models of the honey bee waggle dance and human hand motion, respectively, and in [2] a technique is used to find Motion Description Language (MDL) codes from observed ants.

Our work introduces two specific innovations. The first is in applying system identification to learn the parameters of a decentralized dynamical system from data. The second is in the model structure itself, which uses a novel, non-gradient field to express the interaction between each agent and its environment. We demonstrate the technique by fitting a model to GPS data from a group of three free-ranging cows. The model is validated by testing the whiteness of the residual error, and by comparing global statistics of simulations verse the actual data. We also show how the resulting model can be used to control simple robots. Recently, the technique has been validated with data from ten cows, though we do not present those results here.

2 Model Description

We consider a linear-in-parameters model structure with three naturally distinct parts to describe the motion of coupled physical agents moving over a plane surface. Firstly, each agent is given internal dynamics to enforce the constrains of Newtons laws. Secondly, a force[3] is applied to each agent from its interaction with each of the other agents in the group. Thirdly, a force is applied to each agent as a function of its position in the environment. All remaining effects are modeled as a white noise process.

2.1 Individual Agent Dynamics

Supposing a group of m agents, the proposed model structure for an individual agent $i \in \{1,\ldots,m\}$ can be written in state-space, difference equation form as

$$x_i^{\tau+1} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & a_i & 0 \\ 0 & 0 & 0 & a_i \end{bmatrix} x_i^{\tau} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \left(\sum_{j=1, j\neq i}^{m} f_{ij}(p_i^{\tau}, p_j^{\tau}) + g_i(p_i^{\tau}) + w_i^{\tau} \right). \quad (1)$$

Agent i's state $x_i^{\tau} = [e_i^{\tau} \;\; n_i^{\tau} \;\; u_i^{\tau} \;\; v_i^{\tau}]^T$ consists of its East position, North position, Eastern component of velocity, and Northern component of velocity after the τth iteration, and its position is given by, $p_i^{\tau} = [e_i^{\tau} \;\; n_i^{\tau}]^T$. The time step Δt is given by

[3] In this work the term "force" is used in a metaphoric sense. When we talk of a "force" we are referring to the intention of the agent to accelerate in a particular way using it's own motive mechanisms.

$t^{\tau+1} - t^{\tau}$, and we assume it is constant for all τ. The term a_i represents damping, $a_i = 1$ for zero damping, and $|a_i| < 1$ for stable systems. The function $f_{ij}(p_i^{\tau}, p_j^{\tau})$ determines the coupling force applied by agent j to agent i. The function $g_i(p_i^{\tau})$ represents the force applied by the environment to the agent at point p_i^{τ}. Finally, w_i^{τ} is a zero-mean, stationary, Gaussian white noise process uncorrelated with $p_j \; \forall j$ used to model the unpredictable decision-motive processes of agent i. Nonholonomic constraints which are often present in mobile agents, such as people, cattle, or automobiles, are neglected in this treatment, though they could be incorporated with an increase in the complexity of the model structure. Note that the force terms are only applied to affect changes in velocity in accordance with Newton's second law.

2.2 Agent-to-Agent Interaction Force

Dropping the τ superscripts for clarity, the form of the agent coupling force $f_{ij}(p_i, p_j)$ is given by

$$f_{ij}(p_i, p_j) = \left(\theta_{1_{ij}} - \frac{\theta_{2ij}}{\|p_j - p_i\|} \right) n_{ij}, \tag{2}$$

where $n_{ij} = (p_j - p_i)/\|p_j - p_i\|$ is the unit vector along the line from p_i to p_j (henceforth, $\|\cdot\|$ will denote the ℓ^2 norm). This is the simplest of a family of force laws commonly used in computational models of physical, multi-body systems (e.g. kinetic gas models). The important feature of this family is that an agent is repulsed from its neighbor at close distances and attracted to its neighbor at far distances. To see this property clearly, examine the magnitude of (2) given by $\|f_{ij}\| = \theta_{1_{ij}} - \theta_{2_{ij}}/\|p_j - p_i\|$, and shown in the left of Figure 1.

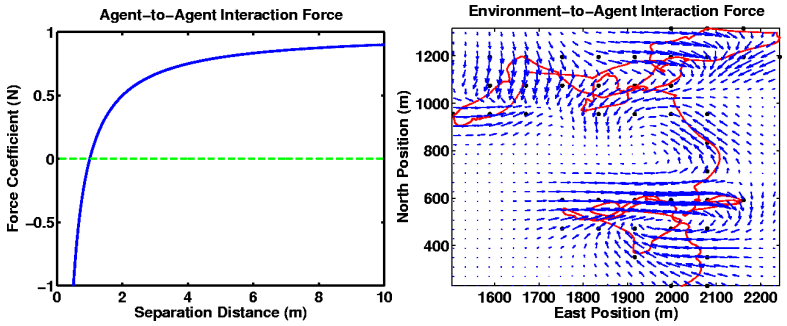

Fig. 1. The magnitude of the agent-to-agent interaction force is shown on the left for $\theta_1 = \theta_2 = 1$. On the right, the vector field representing the force felt by an agent at each point on the plane is shown for an example agent trajectory. The swirling patterns evident in the field are made possible by a novel parameterization.

After some manipulation, the sum of f_{ij} over all neighbors j can be expressed as

$$\sum_{j \neq i} f_{ij} = \begin{bmatrix} \phi_{fu_i} \\ \phi_{fv_i} \end{bmatrix} \theta_{f_i}, \tag{3}$$

where

$$\phi_{fu_i} = \left[\frac{(e_1-e_i)}{\|p_1-p_i\|} \cdots \frac{(e_m-e_i)}{\|p_m-p_i\|} \; \frac{-(e_1-e_i)}{\|p_1-p_i\|^2} \cdots \frac{-(e_m-e_i)}{\|p_m-p_i\|^2} \right]$$
$$\phi_{fv_i} = \left[\frac{(n_1-n_i)}{\|p_1-p_i\|} \cdots \frac{(n_m-n_i)}{\|p_m-p_i\|} \; \frac{-(n_1-n_i)}{\|p_1-p_i\|^2} \cdots \frac{-(n_m-n_i)}{\|p_m-p_i\|^2} \right], \text{ and}$$
$$\theta_{fi} = \left[\theta_{1_{i1}} \cdots \theta_{1_{im}} \; \theta_{2_{i1}} \cdots \theta_{2_{im}} \right]^T,$$

permitting a slight abuse of notation since the indices $j = i$ are excluded from the above vectors. This notation will be useful in what follows.

2.3 Environment-to-Agent Interaction Force

The agent's preference for certain paths in the environment is modeled as a nonlinear mapping from each point on the plane to a force vector felt by the agent. To this end, two networks of Gaussian basis functions are used, one for each of two perpendicular force components.

In particular, the function $g_i(p_i)$ can be written

$$g_i(p_i) = \begin{bmatrix} \theta_{u_{i1}} \cdots \theta_{u_{in}} \\ \theta_{v_{i1}} \cdots \theta_{v_{in}} \end{bmatrix} \begin{bmatrix} \phi_{g_1}(p_i) \\ \vdots \\ \phi_{g_n}(p_i) \end{bmatrix}, \tag{4}$$

where $\phi_{g_{ik}}(p_i) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} \exp(-\frac{\|p_i-\gamma_{ik}\|^2}{2\sigma_{ik}^2})$, and $k \in \{1,\ldots,n\}$. Each Gaussian is centered at γ_{ik}, with standard deviation σ_{ik}, and its strength is represented by the unknown parameters $\theta_{u_{ik}}$ for the Eastern component, and $\theta_{v_{ik}}$ for the Northern component. Gaussian basis functions were chosen for their familiarity; the objective being to demonstrate the modeling approach with a minimum of complications. A number of other basis functions types could be used, including wavelets, sigmoidal functions, or splines.

It is important to note that a vector-field parameterized in this way is *not* a potential gradient. A potential gradient field cannot admit circulation around closed paths.[4] We introduce a non-gradient parameterization to enable circulation, as one can imagine agents intending to traverse closed orbits on the plane. For example, a cow may have a routine of passing between a water source, a shaded tree, and a grassy patch in a periodic fashion.

Figure 1, on the right, shows a plot of an example force-field parameterized in the above way. The arrows show the forces induced by the field, the black dots show the centers of the Gaussian functions, γ_{ik}, and the red curve shows the path of an agent over the vector field. The swirling patterns evident in the vector field would be impossible if it were a gradient field.

[4] proof: Let $\Psi(p)$ be a potential function and $V(p) = -\text{grad}(\Psi)$ its gradient field. Then $\text{curl}(V) = \text{curl}(-\text{grad}(\Psi)) = 0$, thus by Green's Theorem, $\oint_s V ds = \int_{A_s} \text{curl}(V) dA = 0$, where s is any closed curve on the plane, and A_s is the area enclosed by s.

The expression in (4) can be put into a different form to match that of (3). In particular

$$g_i(p_i) = \begin{bmatrix} \phi_{gu_i} \\ \phi_{gv_i} \end{bmatrix} \theta_{g_i}, \tag{5}$$

where

$$\begin{aligned} \phi_{gu_i} &= [\, \phi_{g_1} \ \cdots \ \phi_{g_n} \ \cdots \ 0 \ \cdots], \\ \phi_{gv_i} &= [\, \cdots \ 0 \ \cdots \ \phi_{g_1} \ \cdots \ \phi_{g_n} \,], \text{ and} \\ \theta_{g_i} &= [\, \theta_{u_{i1}} \ \cdots \ \theta_{u_{in}} \ \theta_{v_{i1}} \ \cdots \ \theta_{v_{in}} \,]^T. \end{aligned}$$

Although somewhat awkward, this form will become useful in what follows.

3 System Identification with Least Squares Fitting

The model that was described in Section 2 can be manipulated into a form so that its parameters can be fitted using the Least Squares method for system identification [4]. We assume that only position measurements, p_i^τ, $\tau = 1, ..., N$, are available to perform the fitting. We can eliminate u_i and v_i from the dynamics in (1) to provide a second order equation in the position only. Notice that from (1) we can write

$$p_i^{\tau+1} = p_i^\tau + \Delta t \begin{bmatrix} u_i^\tau \\ v_i^\tau \end{bmatrix}, \tag{6}$$

and

$$\begin{bmatrix} u_i^{\tau+1} \\ v_i^{\tau+1} \end{bmatrix} = a_i \begin{bmatrix} u_i^\tau \\ v_i^\tau \end{bmatrix} + \sum_{j=1, j\neq i}^{m} f_{ij}^\tau + g_i^\tau + w_i^\tau. \tag{7}$$

We can solve (6) for $[u_i^\tau \ \ v_i^\tau]^T$ and substitute into the right hand side of (7). We then substitute the result back into the right hand side of (6), shifting time indices appropriately, to obtain the desired expression

$$p_i^{\tau+2} = p_i^{\tau+1} + (p_i^{\tau+1} - p_i^\tau) a_i + \Delta t \left(\sum_{j=1, j\neq i}^{m} f_{ij}^\tau + g_i^\tau + w_i^\tau \right).$$

We can use the above expression to formulate a one-step-ahead predictor. First, define the combined regressor vectors $\phi_{u_i}^\tau = [\, (e_i^{\tau+1} - e_i^\tau)/\Delta t \ \ \phi_{fu_i}^\tau \ \ \phi_{gu_i}^\tau \,]$, and $\phi_{v_i}^\tau = [\, (n_i^{\tau+1} - n_i^\tau)/\Delta t \ \ \phi_{fv_i}^\tau \ \ \phi_{gv_i}^\tau \,]$, and a combined parameter vector $\theta_i = [\, a_i \ \theta_{f_i}^T \ \ \theta_{g_i}^T \,]^T$. By taking the expectation conditioned on the positions, substituting (3) and (5) for $\sum_{j\neq i} f_{ij}$ and g_i, respectively, then making use of the combined regressor and parameter vectors we get

$$\hat{p}_i^{\tau+2} = p_i^{\tau+1} + \Delta t \begin{bmatrix} \phi_{u_i}^\tau \\ \phi_{v_i}^\tau \end{bmatrix} \theta_i, \tag{8}$$

where $\hat{p}_i^{\tau+2}$ is the expected value of p_i after $\tau+2$ time steps, given positions up to $\tau+1$, and w_i^τ drops out in the conditional expectation.

The Least Squares method is now implemented to find the optimal model parameters. Specifically, we wish to find the parameters, θ_i, to minimize the mean squared prediction error over all available time steps. The mean squared prediction error can be written $J_i = 1/(N-2)\sum_{\tau=1}^{N-2}(p_i^{\tau+2} - \hat{p}_i^{\tau+2})^T(p_i^{\tau+2} - \hat{p}_i^{\tau+2})$. Substituting into J_i with (8) and (6) yields

$$J_i = \frac{\Delta t^2}{N-2}(Y_i - \Phi_i\theta_i)^T(Y_i - \Phi_i\theta_i), \tag{9}$$

where

$$Y_i = [\, u_i^2 \ \ldots\ u_i^{N-1}\ v_i^2 \ \ldots\ v_i^{N-1}\,]^T, \text{ and } \Phi_i = [\, {\phi_{u_i}^1}^T \ \ldots\ {\phi_{u_i}^{N-2}}^T\ {\phi_{v_i}^1}^T \ \ldots\ {\phi_{v_i}^{N-2}}^T\,]^T,$$

and u_i^τ and v_i^τ are obtain from (6). The least squares problem is then formulated as $\theta_i^* = \arg\min_{\theta_i} J_i(\theta_i)$. Following the typical procedure for solving the least squares problem we find that

$$\theta_i^* = [\Phi_i^T\Phi_i]^{-1}\Phi_i^T Y_i. \tag{10}$$

The right hand side of (10) consists entirely of measured data while the left hand side is the vector which represents the optimal parameters of the model. We assume that the data are rich enough that the matrix inversion in (10) is possible. The deep implications of this invertibility are discussed in [4]. The myriad merits and deficiencies of Least Squares fitting compared with other learning methods will not be discussed in this work.

The resulting residual error in the fitting process is now used to define $\hat{w}_i$, so that

$$w_i^\tau = \begin{bmatrix} u_i^{\tau+1} \\ v_i^{\tau+1} \end{bmatrix} - \begin{bmatrix} \phi_{u_i}^\tau \theta_i^* \\ \phi_{v_i}^\tau \theta_i^* \end{bmatrix}.$$

If the "true" system dynamics are represented by the fitted model, we expect to find that w_i is zero-mean, stationary, Gaussian white noise. Specifically, for perfect fitting, $E[w_i(t)w_i^T(t+\tau)] = \delta(\tau)Q_i$, where $\delta(\tau)$ is the Kronecker delta function. Therefore, the "whiteness" of w_i can be used as an indicator of the goodness of fit that has been achieved. For simulation purposes, we would assume w_i is white noise and design Q_i to be equal to the empirical covariance of w_i.

4 Modeling a Group of Cows

The method presented above was used to model the dynamics of a herd of cows. Data collected from actual cows were used for fitting the model parameters and for evaluating the resulting model.

Data were collected in a single trial from three cows wearing GPS devices. For each cow, data consisted of approximately 3100 GPS position entries collected at 1Hz. The data for all animals were artificially synchronized to a common clock using a standard linear interpolation. The characteristic time scale of cow dynamics is considerably longer than 1 second, thus such an interpolation is expected to have a negligible effect on modeling results.

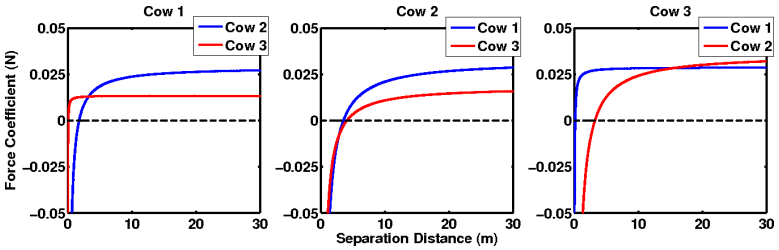

Fig. 2. The agent-to-agent interaction forces are shown for the three cows. Each curve represents the size of the force imposed by one cow on another as a function of the distance between the cows. A positive value is attractive while a negative value is repulsive.

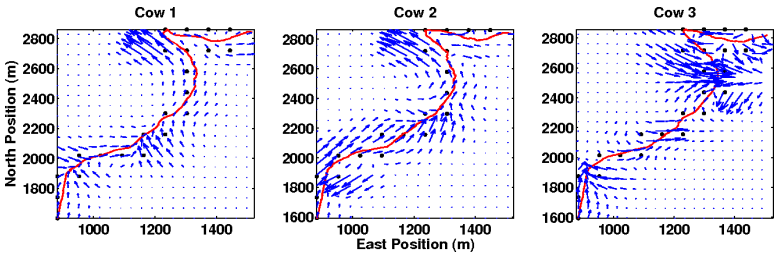

Fig. 3. The environment-to-agent force field is shown for the three cows. The black dots are the centers of the Gaussian functions and the red curve shows the data used for regression, marking the cow's path over the region.

4.1 Least Square Results

The data were used to find model parameters as described in section 3. The panels in Figure 2 show the force coefficients $F_{ij}(p_i, p_j)$ for the three cows. For each cow the two plots show the interaction force with each of the two other cows in the herd. Note that the forces are not necessarily pair-wise symmetric, that is, $F_{ij} \neq F_{ji}$ in general.

The environment-to-agent vector fields are shown in the panels of Figure 3 for the three cows. The black dots show the centers of the Gaussian basis functions, γ_{ki}, the blue arrows show the direction and magnitude of the force felt by a cow at each point, and the red dots show the position data used for regression. The Gaussian centers were spaced over an even grid containing the trajectory of the cow. If the trajectory did not enter the grid region of a Gaussian function, it was dropped from the network. This primitive pruning algorithm was used for simplicity; more complex algorithms could be employed. The Gaussian widths were chosen to be 2/3 the length of the grid space occupied by the Gaussian.

The ability of the model to capture the relevant qualities of the data was investigated by examining the covariance function of the residual error, w_i, in comparison with the covariance function of the velocity as calculated by (6). The top of Figure 4 shows the empirically calculated auto-covariance of the Eastern component of the velocity for Cow 1, and the bottom figure shows the empirically calculated auto-covariance of the corresponding residual error. The dotted lines indicate a 90%

whiteness confidence interval. This means that, for each point on the auto-covariance function, a stationary Gaussian white noise process would have produced an empirical auto-covariance within the dotted lines with probability .9. Notice that the strong temporal correlation in the velocity has been greatly reduced in the residual, indicating that the model has captured a good deal of the predictable qualities of the data. Furthermore, a large majority of the residual auto-covariance points lie inside the confidence interval, giving good reason to believe that it is a "nearly white" signal. The plots for the other cows are excluded in the interests of space, though the results are similar.

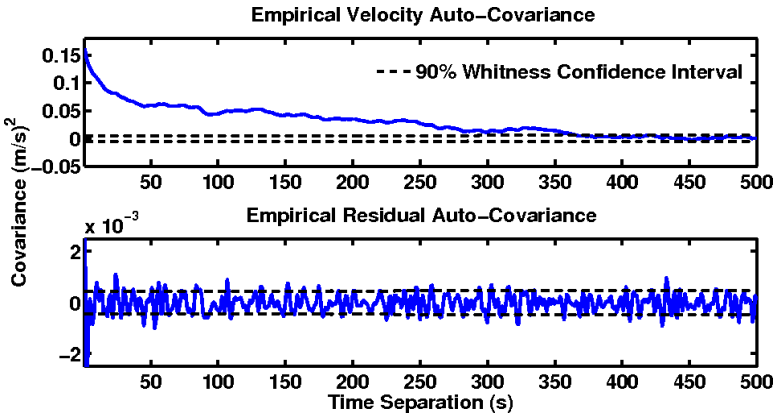

Fig. 4. The empirical covariance function for the Eastern component of the velocity is shown in the top plot, and for the error residual in the bottom plot. The dotted lines indicate a 90% whiteness confidence interval, meaning that a Gaussian white noise process would have generated an empirical auto-covariance inside the interval with probability .9 at each point. Clearly the residual error is "more white" than the original velocity.

4.2 Herd Simulation

Simulation experiments were carried out with the models fitted in Section 4.1. We simulated a group of three simple mobile robots controlled to have the dynamics in (1) with the parameters found in section 4.1. The difference equations were numerically solved in a Matlab environment.

The trajectories of the robots from a typical simulation are shown in the left side of Figure 5 laid over a schematic showing the fences of the paddock where the actual cow data were recorded. The trajectories of the simulation are similar to those of the real cows. What is perhaps more surprising is that the simulated robots track the fence lines, as did the real cows. This tendency is captured solely through the agent-to-environment force field (described in section 2.3), as the model has no direct knowledge of where fence lines may lie. Furthermore, statistics were gathered for the simulated robots and compared with those from the cow data. Figure 6 shows

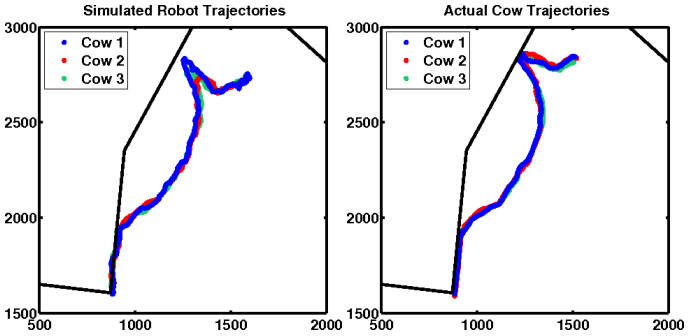

Fig. 5. The left picture shows trajectories of a team of simulated robots controlled to behave like a herd of cows. The robots use dynamical laws generated from the procedure described in this work. Their positions are laid over the fence lines of the paddock where the original data were collected, though they have no direct knowledge of fence positions. The right picture shows the actual cow data over the same time window.

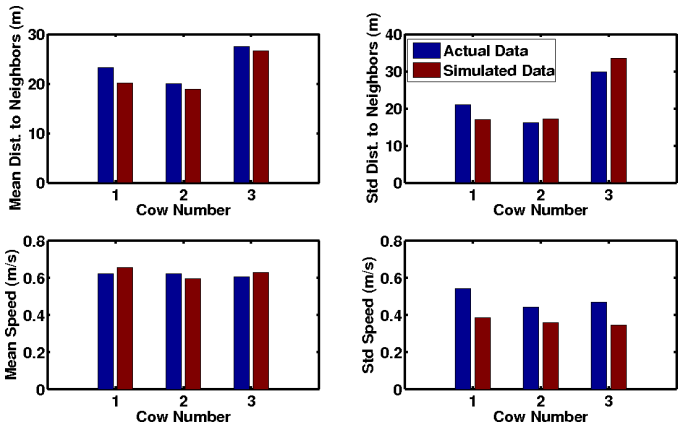

Fig. 6. The bar charts compare various statistics for the actual cow data and the simulated robots. The top charts show the mean and standard deviation of the distance from one cow to the other two cows in the group. The bottom charts show the mean and standard deviation of the speed of each cow.

a comparison of the two sets of statistics. Notice that global properties as measured by these statistics are preserved by the model.

5 Conclusions

In this paper we presented a method to generate models of groups of dynamical agents (e.g. flocks, herds, and swarms) using observations of the agents' positions over time. We formulated a physically motivated difference equation model, and used the Least Squares system identification method to fit the model to data from a

herd of three cows. Many avenues remain open for the extension of this work. Most importantly, the method should be tested on data from much larger groups, though such data is difficult to collect. Recently, we have validated the technique with data from ten cows with results very similar to those presented here, however data from an order of magnitude more animals would likely reveal collective behaviors not seen in smaller herds. It would also be interesting to compare this model structure with others, to use different learning algorithms, to explore recursive learning algorithms for on-line learning, or to examine analytical properties of the learned models.

6 ACKNOWLEDGMENTS

This work was done in the Distributed Robotics Laboratory (DRL) at CSAIL-MIT and was supported in part by NSF grant IIS-0513628 and the MURI SWARMS project. We are grateful for this support. We thank the DRL lab members for stimulating feedback and discussions. We thank Iuliu Vasilescu and to Carrick Detweiler for designing and building the hardware used to collect the animal data. The cattle data was collected in collaboration with Dr. Dean Anderson on the Jornada Experimental Range under the New Mexico State University Institutional Animal Care and Use Committee (IACUC), Number 2007-001. We thank the staff at the US Department of Agriculture - Agricultural Research Service, Jornada Experimental Range (USDA-ARS-JER) for their support in using their facilities and animals.

References

1. N. Correll and A. Martinoli. System identification of self-organizing robotic swarms. In *Proceedings of 8th Int. Symp. on Distributed Autonomous Robotic Systems*, Minneapolis/St. Paul, Minnesota, USA, July 12-14 2006.
2. F. Delmotte, M. Egerstedt, and A. Austin. Data-driven generation of low-complexity control programs. *International Journal of Hybrid Systems*, 4(1&2):53–72, March & June 2004.
3. A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, June 2003.
4. L. Ljung. *System Identification: Theory for the User*. Prentice-Hall, Upper Saddle River, New Jersey, 1999.
5. S. M. Oh, J. M. Rehg, T. Balch, and F. Dellaert. Data-driven MCMC for learning and inference in switching linear dynamic systems. In *Proceedings of 20th National Conference on Artificial Intelligence*, Pittsburgh, USA, 2005.
6. V. Pavlovic, J. M. Rehg, and J. MacCormick. *Advances in Neural Information Processing Systems 13 (NIPS*2000)*, chapter Learning Switching Linear Models of Human Motion. MIT Press, 2001.
7. C. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21:25–34, 1987.