Modeling Temporally Dynamic Environments for Persistent Autonomous Agents

Matthew J. O'Brien, Ronald C. Arkin
Mobile Robot Lab, Georgia Tech, Atlanta GA 30308
{mjobrien,arkin}@gatech.edu

Abstract

This paper explores how an autonomous agent can model dynamic environments and use that knowledge to improve its behavior. This capability is of particular importance for persistent agents, or long-term autonomy. Inspiration is drawn from circadian rhythms in nature, which drive periodic behavior in many organisms. In our approach, the chemical oscillators from nature are replaced with methods from time series analysis designed for forecasting complex season patterns. This model is incorporated into a behavior-based architecture as an advanced-percept, providing future estimates of the environment rather than current measurements. A simulated application of a janitor robot working in an environment with heavy pedestrian traffic was created as a testbed. Experimental data used real world pedestrian traffic counts and showed an agent using online forecasting of future traffic outperformed both a reactive, sensor-based, strategy and a strategy with a deterministic schedule.

Introduction

The field of robotics has seen great success in industrial environments, and new advancements are continuing to push the boundaries of the field. Despite this, temporally dynamic environments are still very challenging for autonomous systems. This paper explores explicitly modeling the state of a dynamic environment so an autonomous system can predict future changes, and respond proactively.

As with many robotic systems, the inspiration for the approach comes from biology. The natural world contains constant environmental changes, the most significant being solar and seasonal cycles. It is not surprising that animals have adapted to vary their behavior with these cycles (e.g. sleep and migration). As with many bioinspired systems, mimicry is not the goal. The implementation of this system uses time series analysis to provide a mathematically sound foundation for modelling environmental changes. This allows the system to model more than just periodic cycles, including short-term trends and other statistical correlations between measurements. The model is treated as an advanced-percept in a behavior-based architecture, transforming current and past measurements to create an estimate of the future. The resulting vector of future values can be passed to individual behaviors as parameters, or to coordination mechanisms for action-selection.

This approach is a type of data-driven modelling. The physics that determines the state of the environment is, in many cases, far too complex to model directly. Most mammals know nothing about the orbital mechanics that drive the day/night and seasonal cycles. Human traffic follows clear patterns, but the processes that generate it are impossible to measure. Nature has already demonstrated that an organism can gain value from having simple models (chemical oscillators) of complex processes (tides, seasons, etc.) in the environment.

Related Work

Circadian systems

Many organisms in nature exhibit rhythmic behavior. Some of these cycles are not merely responses to periodically changing inputs from the environment, but come from internal processes of the organisms (Aschoff 1981). This allows these animals to take the right actions at the right times, before their needs (internal state) or senses (perceptual state) could inform their behavior.

> Circadian rhythmicity of behavior represents an animal's information, or one is tempted to say, knowledge about a particular feature of its environment… and what to do about it. (Oatley 1974)

A circadian clock has three primary components. An internal oscillator which tracks the passage of time and defines the approximate period of a cycle. The sensory input

* Research supported by Office of Naval Research under Grant #N00014-15-1-2115

channel which allows the oscillator to entrain to the environment. Similar to a mechanical oscillator, entrainment drives the circadian oscillator at the relevant environmental frequency, and ensures it will resync even after extreme disturbances (Roenneberg, Daan, and Merrow 2003). Finally, an output channel influences the agent's behavior based on the state of the oscillator. This simple structure is shown in Figure 1.

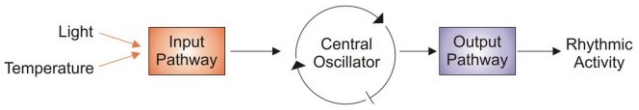

Figure 1: Basic circadian system diagram (Gardner et al. 2006)

Long-Term Autonomy

Recent research in long-term autonomy has pointed attention towards several challenging problems, including reasoning about dynamic environments. Particular focus has been put into the areas of simultaneous localization and mapping (SLAM) and path planning. Explicitly modelling and predicting the changing environment, rather than merely compensating for it, has only recently begun being explored.

An approach for scene recognition over seasons created a visual dictionary that linked corresponding superpixels from summer and winter images, treating seasons as a binary variable (Neubert, Sünderhauf, and Protzel 2015). Path planning through a dynamic environment has been demonstrated by modelling the dynamic state of an edge in a graph as a stochastic process, and solving for a path with minimum expected travel time using dynamic programming (Lobli, Meyer-Delius, and Pfaff 2013).

(Krajnik et al. 2016) covers an approach to model binary environmental states as a probability, varying according to a Fourier based periodic series. This exploits the periodic nature of many environmental changes explicitly. The concept has been applied to path planning (modelling traversabilitiy of edges), localization (presence of specific features) and information driven exploration (entropy of state).

Approach

The Model

In this work the relevant environment state variables are stored as a time series. A fundamental approach to modeling time series is the classical decomposition, which separates a time series into three relevant components.

$$Y_t = T_t + S_t + E_t \qquad (1)$$

Y_t is the original time series. T_t is the trend component, a slowly changing average level. S_t is the seasonal component, a repeating pattern with known period. Finally, E_t is the residual, error, or values left over after the trend and seasonal components have been removed from the time series. This decomposition provides a structured approach for modelling and allows for focus on components of interest. For example, seasonal effects are sometimes removed from data in a process called deseasonalization. In this work, the seasonal effects are of key importance.

This paper employs the TBATS (Trigonometric, Box-Cox transform, ARMA errors, Trend, and Season Componenets) forecasting model framework (De Livera et al. 2011); the basic model is shown below. The time series, y_t, is first transformed using the inverse hyperbolic sine (2a). This allows some non-linearities to be modelled by this linear approach. The series y_t^T is decomposed into the summation of four components (2b). The first two components, l_t (2c) and b_t *(2d),* are the level (or current value) and trend (or current rate of change) of the series after the seasonal terms s_t are removed. These components implement a dampened version of Holt's linear trend method. This technique comes from a family of approaches known as exponential smoothing, which apply an exponentially weighted moving average over past data to predict future values. For a thorough description of exponential smoothing see (Hyndman et al. 2008).

$$y_t^T = \sinh^{-1} y_t \qquad (2a)$$

$$y_t^T = l_{t-1} + \phi b_{t-1} + \sum_i s_{t-1}^i + d_t \qquad (2b)$$

$$l_t = l_{t-1} + \phi b_{t-1} + \alpha d_t \qquad (2c)$$

$$b_t = (1-\phi)b + \phi b_{t-1} + \beta d_t \qquad (2d)$$

$$s_t^i = \sum_{j=1}^{k_i} s_{j,t}^i \qquad (2e)$$

$$d_t = \sum_{i=1}^{p} \varphi_i d_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t \qquad (2f)$$

The component s_t^i is the ith seasonal component (2e). Each seasonal component represents the average values of a series over a certain period. It is efficiently described as a sum of sin and cosine terms, also known as a Fourier series, with k harmonics described in equations (3a) and (3b). All seasonal components are treated independently, with their effects summed together in equation (2b). For the experiment described later in this paper, only one seasonal component was ever used.

$$s_{j,t}^i = s_{j,t-1}^i cos\lambda_j^i + s_{j,t-1}^{*i} sin\lambda_j^i + \gamma_1^i d_t \qquad (3a)$$

$$s_{j,t}^{*i} = -s_{j,t-1}^i sin\lambda_j^i + s_{j,t-1}^{*i} cos\lambda_j^i + \gamma_2^i d_t \qquad (3b)$$

The final component, d_t, is an ARMA(p,q) process to model any residual error (2f). ARMA (autoregressive moving average) models are able to accurately approximate a

large class of stationary time series. Extensions to non-stationary time series can be done by through a decomposition and removal of the trend and seasonal (time-varying) components leaving the stationary residual (as done here). Another option is to difference the time series with itself. Many types of non-stationary time series reduce to ARMA processes after a finite number of differences. These are referred to as ARIMA models (autoregressive integrated moving-average). For a thorough introduction to using ARMA models for modelling and forecasting, see (Brockwell and Davis 2006).

The Akaike information criterion (AIC) is used for model selection (e.g. the order of the ARMA(p,q) process). While the parameters themselves are estimated with a combination of maximum likelihood estimation and numerical optimization (the details are out-of-scope for this paper, see the original paper for more information). These methods allow both model selection and parameter estimation to be automated. This is a useful property for robotic systems. If a robot is moved to a new environment, or the environment changes drastically, the model can be updated without the need for expert intervention.

The forecast package for R[1] contains an implementation of the TBATS model used in this work.

Testbed – Janitor Robot

To explore and test the ideas discussed, an example scenario was created for an autonomous janitorial robot. This robot attempts to keep a public and high traffic area clean. The issue is that heavy volumes of pedestrian traffic both inhibit the ability of the robot to move, and the presence of the robot interferes with the pedestrian traffic. Ideally the robot can return to a charging station when traffic is high, and clean whenever traffic is low.

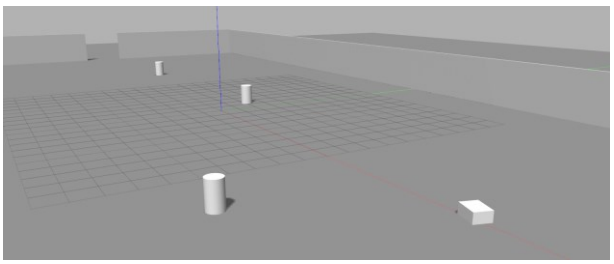

Figure 2: Gazebo simulation environment. The cylinders represent pedestrians, and the box represent the robot.

The simulated environment was a 60m by 26m rectangular area, walled off, with two entrances/exits in the center of the short sides (see Figure 2). Pedestrians used simple state machines to enter the park at one entrance, wander around for a brief amount of time, and exit at the other side. Pedestrians used potential-fields based obstacle avoidance to keep from colliding with other pedestrians or the robot. The robots charging station was located in one corner of this area.

Real pedestrian traffic data was used for both modeling and simulation. Arlington, VA has setup a system of pedestrian and bike counters and provides access to the data online[2]. This provided traffic counts with resolution up to 15 minute intervals. Some example days are shown below in Figure 3. To simulate real-time traffic from these counts, start times for each pedestrian were randomly distributed within each 15-minute time interval using a uniform distribution.

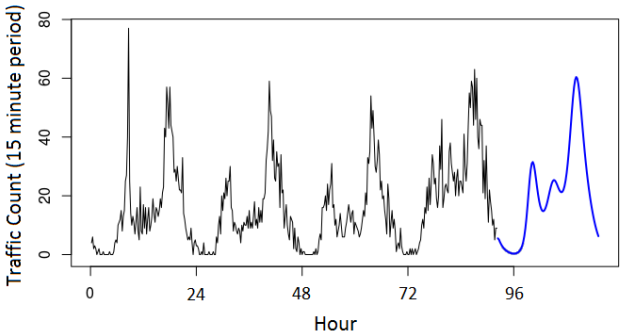

Figure 3: Four days of traffic counts (black) followed by one day of forecasted traffic values from TBATS (blue).

No actual cleaning or charging actions were simulated. For cleaning, the robot performed a random walk. The time spent in the cleaning state was taken as the measure of cleaning done. For charging, the robot moved to the designated area and waited. There were no power constraints, the robot could clean indefinitely. Despite this simplification, it's still desirable for the robot to return to the base charger during periods of high traffic in order to avoid interfering with pedestrians. As this was exploratory work, these real-world issues were simplified to focus on the interaction between the robotic agent and traffic.

Input Pathway

The first step required is to initialize the system with a set of data sufficient to generate an accurate model. Multiple periods of any relevant seasonal cycles must be provided. Two work weeks (September 7-11 and 14-18, 2015) were used to create a traffic model for a work day.

Once the robotic system is running, it measures the traffic count itself by tallying every unique pedestrian viewed in simulation (see section 'Sensor Model' below for details). Every fifteen minutes (one forecast interval) the robot includes the new data point and produces a new set of forecasts. At this time, the robot can either generate new forecasts using the new data and the same model, or it can

[1] http://robjhyndman.com/software/forecast/

[2] http://www.bikearlington.com/pages/biking-in-arlington/counting-bikes-to-plan-for-bikes/

re-calculate the model and parameters first. When and how often to update the model is an open question. In this work the model and parameters were not recalculated.

Output Pathway

The model shown in equations (2a-e) takes in recent measurements of a time series and produces future predictions of that time series. In this application, the forecast returns expected counts for 15-minute time periods, *T*. The length of T is determined by the input data. What the robot needs is the expected number of people at any instant in time. To achieve this, view the forecast at time *t* as the count over a range of time from *t-T/2* to *t+T/2*. There is only one estimate every fifteen minutes. Between these times, a simple first-order approximation can be used. Figure 4 below demonstrates this process. Detection of pedestrians (red dashes) are recorded and used to generate counts for the past (red circles). Together with the predicted counts for the future (blue circles), a predicted count during a 15-minute interval around *any* time *t* can be calculated.

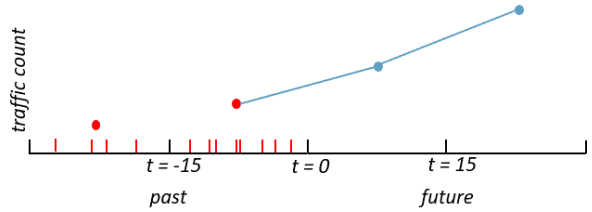

Figure 4: Predicting traffic counts at any time based on past values (red) and forecasted future values (blue). Red dashes are individual detections; red circles are the final count per period

To convert this forecasted count into a density (i.e. the number of pedestrians in the area at some instant in time) the average time spent by each pedestrian in the relevant area, T_{ped}, is needed. Given *F(t)* returns the forecast at time *t* as calculated above, then the density at time *t* is…

$$Density(t) = F(t) * \frac{T_{ped}}{T} \qquad (4)$$

If there is significant variation in the average time spent, the real density could vary significantly from the estimated density, even with perfect forecasting of the counts. Another option is to simply forecast the density directly with measurements of the number of pedestrians at given points in time. For this experiment density data was not available, only counts, so this was not possible.

Behavior

With an estimate of the traffic density for any point in time in the future, we can look at how it can be incorporated into behavior and decision-making. For this experiment, a behavior-based architecture utilizing motor schema was applied (Arkin 1998). A basic finite state automata (FSA) transitioned the robot between three behaviors: *cleaning*, *returning*, and *waiting*. The *cleaning* state was a random search, the *returning* state drove the robot to its home-base, and both behaviors included obstacle avoidance to evade pedestrians.

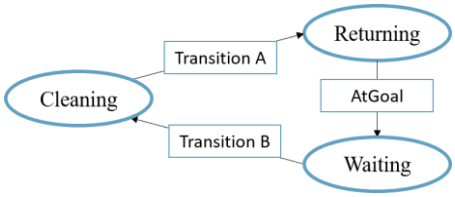

Figure 5: FSA for janitor robot

A purely reactive agent may transition from *cleaning* to *returning* (Transition A) when it sees the traffic density has crossed some threshold. The forecasting agent can instead transition when it forecasts that at some time limit in the future, T_{lim}, this threshold will be crossed. T_{lim} is chosen to ensure that the robot has enough time to return to its station. In this way, the time series model can be treated as an advanced-percept. It returns the same type of information about the environmental state as real-time sensors do, but future values instead of current measurements.

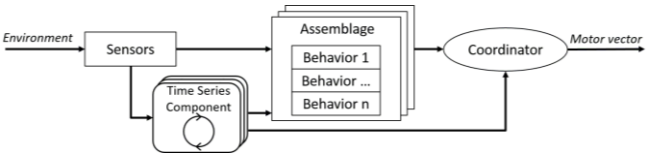

Figure 6: In a traditional behavior-based architecture, sensor measurements feed directly into the relevant behaviors. Here, the TBATS model is treated as another percept in parallel with other sensor measurements.

This implementation transitioned from *cleaning* to *returning* (Transition A) when two conditions were met:

$$Density(T_{lim}) > Threshold$$
$$Density(2T_{lim}) > Threshold$$

The second check for the time T_{lim} in the future is simply to ensure the increase in traffic is long enough to be meaningful. The conditions for transitioning from *waiting* to *cleaning* (Transition B) were:

$$Density(T_{lim}) < Threshold$$
$$CurrentTraffic() < Threshold$$

This ensures the robot returns to cleaning when the traffic is expected to be dropping below the threshold, but also waits to ensure the traffic is actually clear.

Experiment

Competing Strategies

In addition to the forecasting procedure discussed above, two other strategies for the janitorial robot were tested for comparison. A reactive strategy only utilized real-time

sensing. Transitions A and B happened when the *current* number of pedestrians was above or below the specified traffic density threshold.

A schedule-based strategy utilized the seasonal component of the TBATS model to find an average traffic prediction, and hard-coded start and stop times based on this average. All strategies used the same threshold for traffic density, which was an arbitrarily chosen value near the median value for an average day.

Sensor Models

Two sensor models were tested to examine the performance of the above approaches with varying amounts of information. A global sensor model provided the robot with the count of all pedestrians in the park. Real scenarios could utilize environmental sensing to track the presence of pedestrians, and communicate that to a robot. Consider a subway station which counts individuals using a turnstile, and relays that count to an autonomous cleaning robot.

A local sensor model provided the position of all pedestrians within 20 meters. This simulates the limits of onboard sensing. These were applied to the reactive and forecasting agents. The schedule-based agent did not utilize sensing beyond short-range obstacle-avoidance, so there would be no difference with either sensor model.

Dependent Variables

The goal of the janitorial robot is to maximize cleaning while minimizing traffic interference. The time the robot spent in each state (cleaning, returning, waiting) was recorded for every 15-minute interval. The time spent in the cleaning state is treated as a stand-in for the cleaning performance. Traffic interference was measured by recording the number of unique "traffic-disruptions". Anytime a pedestrian came within two meters of a robot, it was counted as a single traffic-disruption. Two meters is also the distance that the robot's obstacle avoidance behaviors began functioning.

Strategies that clean less will typically have less disruptions, and vice versa. This makes quantitatively comparing strategies difficult without assigning utility values and costs to cleaning and disruptions. This issue can be sidestepped by comparing performance to the intended performance defined by the traffic threshold. *T-Disrupting* is defined as the time the robot spent cleaning or returning during 15-minute intervals where the traffic count was above the limit. In contrast, *T-Wasted* is the time spent in the waiting state during intervals where the traffic count was below the limit. Finally, *Total-T-Off* is the sum of *T-Wasted* and *T-Disrupting*, which measures the total deviation from ideal behavior.

Experiment Details

Three days were tested, using pedestrian count data from September 21, 22, and October 19. Two immediately following the days used to initialize the model, and one day four weeks later to see if potential variation over longer time effected the model. In the middle of night, with little to no traffic, all strategies selected to always clean. The time period from 11pm to 5am was excluded for this reason.

Results

The daily results of each strategy, averaged for all three days, are shown below in Table 1. In general, the schedule strategy was conservative, with few disruptions and almost no time cleaning during periods of high traffic, despite varying traffic for each day. This also meant the total time spent cleaning was the lowest of any strategy, and both *T-Wasted* and *Total-T-Off* were the highest of any strategy.

The reactive strategy performance varied significantly between sensor models. With a global sensor, it was competitive with the forecast strategy for the best total performance. When reduced to using local sensing, however, the performance of the reactive system dropped significantly. While it had almost no *T-Wasted*, it spent a significant amount of time cleaning during periods of high traffic, causing the most traffic disruptions. This approach was greedy and would set out to clean during random short gaps in traffic.

The forecast strategy with global sensing had essentially equal performance to the reactive strategy with global sensing. With local sensing, however, the forecast strategy's performance only dropped slightly. The forecasting agent needed only to see a pedestrian once to include them in its count, and maintain an accurate model to base its actions on. Few pedestrians were missed entirely using the local sensor model, even if the robotic agent couldn't sense most pedestrians at the same time.

Figure 7 presents the time spent cleaning over the course of a day, when using local sensing, and compares it to the traffic level and number of disruptions created. One day for each strategy is shown. It is clear that the schedule strategy missed cleaning during periods of low-traffic, and the reactive strategy continued to clean some amount during high traffic. The forecast strategy was better able to avoid both of these situations.

Table 1: Results, three-day-average

	Schedule	Reactive (global)	Reactive (local)	Forecast (global)	Forecast (local)
Disruptions	68.3	212.7	366.7	238	238.33
T-Cleaning (m)	349.7	591.5	838.7	591.9	600.3
T-Disrupting (m)	6.93	79.1	312.2	86.1	106.5
T-Wasted (m)	339.8	112.8	0.387	108.2	103.7
Total-T-Off (m)	346.8	191.9	312.6	194.3	210.2

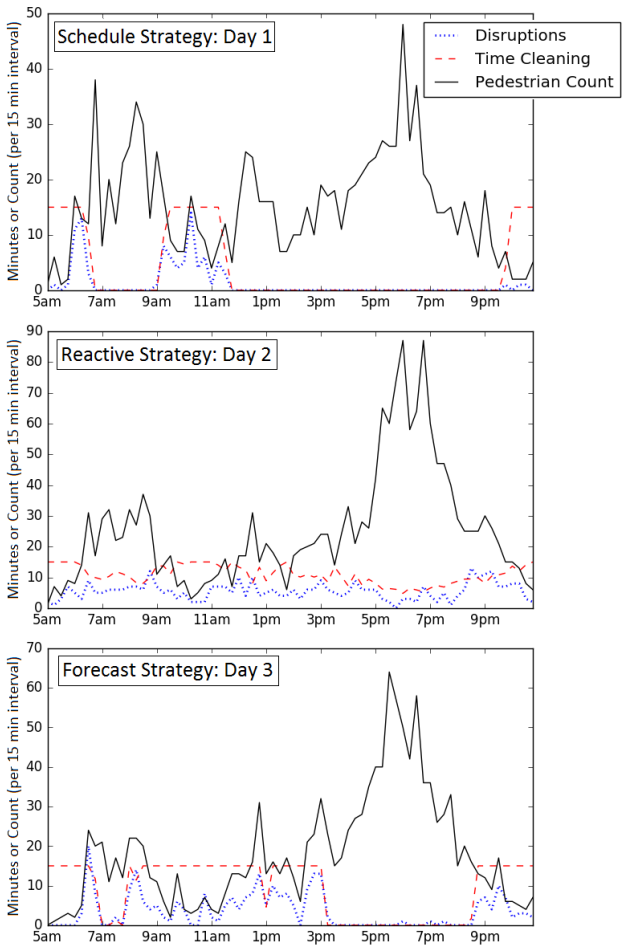

Figure 7: Example days for each strategy using local sensing. Each point is the total traffic count, number of disruptions, or minutes spent cleaning over a 15 minute interval.

Discussion

The reactive and schedule based strategies represent extremes. If the environment is purely random, there is no way to predict future values, and a reactive approach is the best that can be done. Likewise, if the environment is perfectly deterministic, then a hard schedule can provide the exact future states. Many interesting cases, like the traffic patterns used in this work, lie in-between these extremes. There are both clear patterns and significant random variation. It is these in-between cases where forecasting environmental states should provide improved performance.

This approach is far from perfected. In particular, the time series model applied in this paper is fundamentally discrete, designed for use off-line on sets of data. This leaves the forecasting system operating at a delay. Observe the forecast strategy graph in Figure 7 at 1pm. A temporary spike in traffic caused the predictions to be adjusted, and cleaning was stopped temporarily. The short spike in traffic, however, was gone before the agent updated its behavior. This response could be improved either with more low-level reactive behaviors to deal with immediate concerns, or more sophisticated time series models that can update continuously.

There is also substantial opportunity to explore how to utilize these predictions to drive behavior. In this initial work the task and environment were simple, leading to an intuitive implementation. When the robot's actions also impact the state (such as draining power or removing dirt) the problem becomes far more complicated. Approaches using reinforcement learning to solve this complete problem are currently being explored. Regardless, the work here indicates that even simply approaches can improve the performance of an autonomous agent. Many robotic applications that deal with dynamic environments could see benefits from maintaining explicit models of how the environment changes, and taking actions based on the future state.

References

Arkin, R. C. 1998. *Behavior-based robotics*. Mass.: MIT press.

Aschoff, J. 1981. A Survey on Biological Rhythms. In *Handbook of Behavioral Neurobiology.* Vol.4: *Biological Rhythms,* 563-571. New York: Plenum Press.

Brockwell, P. J., and Davis, R. A., eds. 2006. *Introduction to Time Series and Forecasting.* New York: Springer

De Livera, A. M.; Hyndman, R. J.; and Snyder, R. D.; 2011. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association* 106(496): 1513-1527.

Gardner, M. J.; Hubbard, K. E.; Hotta, C. T.; Dodd, A. N.; and Webb, A. A. 2006. How plants tell the time. *Biochemical Journal* 397(1): 15-24.

Hyndman, R., Koehler, A. B., Ord, J. K., and Snyder, R. D., eds. 2008. *Forecasting with exponential smoothing: the state space approach.* Berlin: Springer-Verlag

Krajnik, T.; Fentanes, J. P.; Santos, J. M.; and Duckett, T. 2016. Frequency map enhancement: introducing dynamics into static environment models. Paper presented at ICRA 2016 Workshop on AI for Long-term Autonomy, Stockholm, Sweden, 16 May

Lobli, S.; Meyer-Delius, D.; and Pfaff, P. 2013. Probabilistic time-dependent models for mobile robot path planning in changing environments. In Proceedings of the IEEE International Conference on Robotics and Automation, 5545-5550. Karlsruhe, Germany: International Conference on Robotics and Automation

Neubert, P.; Sünderhauf, N.; and Protzel, P. 2015. Superpixel-based appearance change prediction for long-term navigation across seasons. *Robotics and Autonomous Systems* 69: 15-27.

Oatley, K. 1974. Circadian Rhythms and Representations of the Environment in Motivational Systems. In *Motivational control systems analysis*, ed. D. J. McFarland, 427-459. Oxford: Academic Press.

Roenneberg, T.; Daan, S.; and Merrow, M. 2003. The art of entrainment. *Journal of Biological Rhythms* 18(3): 183-194.