Building Dynamic Cloud Maps from the Ground Up

Calvin Murdock
Machine Learning Department
Carnegie Mellon University
cmurdock@cs.cmu.edu

Nathan Jacobs
Department of Computer Science
University of Kentucky
jacobs@cs.uky.edu

Robert Pless
Department of Computer Science and Engineering
Washington University in St. Louis
pless@cse.wustl.edu

Abstract

Satellite imagery of cloud cover is extremely important for understanding and predicting weather. We demonstrate how this imagery can be constructed "from the ground up" without requiring expensive geo-stationary satellites. This is accomplished through a novel approach to approximate continental-scale cloud maps using only ground-level imagery from publicly-available webcams. We collected a year's worth of satellite data and simultaneously-captured, geo-located outdoor webcam images from 4388 sparsely distributed cameras across the continental USA. The satellite data is used to train a dynamic model of cloud motion alongside 4388 regression models (one for each camera) to relate ground-level webcam data to the satellite data at the camera's location. This novel application of large-scale computer vision to meteorology and remote sensing is enabled by a smoothed, hierarchically-regularized dynamic texture model whose system dynamics are driven to remain consistent with measurements from the geo-located webcams. We show that our hierarchical model is better able to incorporate sparse webcam measurements resulting in more accurate cloud maps in comparison to a standard dynamic textures implementation. Finally, we demonstrate that our model can be successfully applied to other natural image sequences from the DynTex database, suggesting a broader applicability of our method.

1. Introduction

Large-scale maps of cloud cover are an integral component in weather prediction and analysis. While they are usually obtained by a small number of expensive government satellites, recent work [16] has demonstrated the potential for leveraging geo-located outdoor webcams as an alternative source of this valuable information. Thus, instead of obtaining a single view of clouds from above, a vast system of ground-based local webcam measurements can be coordinated to produce a similar global representation, as shown in Figure 1. While [16] used only spatial interpolation methods to combine local measurements of cloudiness, we demonstrate that the synthesis performance can be improved by incorporating temporal constraints by modeling the regular dynamics of cloud motion. To demonstrate this, we train our model on one year of data captured from 4388 webcams every half hour and test on six months of extrapolated data. To the best of the authors' knowledge, this is the

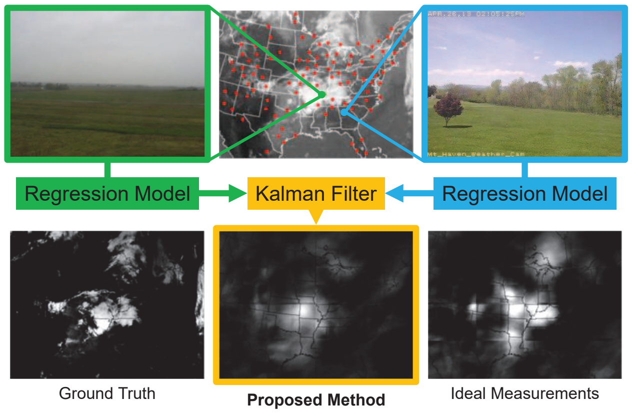


Figure 1: We construct large-scale maps of cloud cover by modeling cloud motion through a linear dynamical system that is Kalman filtered to integrate noisy, sparse measurements acquired from regression models on webcam imagery. Our model is trained on one year of data from 4388 webcams alongside concurrent satellite imagery. Even with significant measurement noise and uncertainty, our model produces reasonable alternatives to satellite imagery. To decouple the challenge of modeling cloud dynamics from acquiring cloudiness measurements from webcams, we also compare against synthesized images using ideal measurements sampled from ground truth data.

largest-scale application of webcam imagery both in terms of temporal extent and number of data sources.

The framework of dynamic textures is a family of approaches commonly used to model image sequences of dynamic, real-world phenomena such as flames, smoke, and waves. However, like clouds, many sequences contain more complicated variations and are unable to be accurately represented using ordinary dynamic textures. In this paper, we introduce an extension that is motivated by the task of modeling complicated cloud motion. We show that our formulation improves representational power while allowing for larger models with many variables. In addition, our formulation is general and can be applied to a variety of other difficult sequences with improved modeling performance.

Ordinary dynamic textures rely on a linear basis of appearance and are well-suited to modeling changes that are correlated across the extent of an image. However, many scenes exhibit variations that are only correlated within smaller spatial neighborhoods. When represented by an appearance basis across the entire scene, the interactions within the dynamical model are complex, leading to model instability, poor predictive performance, and dramatically larger training data requirements. This is especially prominent in the extensive variation present in large-scale cloud distributions. To address this issue, we introduce a hierarchical model that captures the dynamics of appearance at different spatial scales. The resulting dynamics are regularized to be consistent with large-scale variations, but independent of the small-scale variations in non-overlapping regions. Intuitively, this better models multi-scale correlations including weather fronts that span hundreds of miles along with smaller, localized clouds. Defining an overall system requires a large number of variables, but with less interaction between variables and fewer free parameters, overfitting and training data requirements are reduced.

The dynamics of this sparse hierarchical structure can still be expressed as a linear transformation between time steps. Thus, it can be used as the state transition matrix of a Kalman filter to allow partial, noisy measurements of a true state to drive the image synthesis process. In the case of satellite imagery, these can be attained from regression on webcam imagery as described in [16]. Furthermore, this framework allows us to effectively enforce consistency between neighboring pixels at region boundaries where appearances should be correlated but would otherwise be modeled independently by the hierarchical model. Across these boundaries, we show how to enforce *a priori* smoothness constraints on pixels in non-overlapping regions as implicit measurements that can also be included in the Kalman filter. Effectively, this loosens the independence assumption and allows for limited interaction between neighboring regions.

Our contributions within this paper are: (1) the first derivation of a hierarchical dynamic texture model, (2) a description of how to integrate that model within a Kalman filter and allow for explicit sparse measurements to drive the texture synthesis process along with implicit constraints to address smoothness at the boundaries of the hierarchy, (3) an application to continental scale satellite cloud mapping that shows the potential benefits of driving a large scale dynamic texture model with sparse measurements, and (4) extensive testing across a large collection of DynTex [17] sequences demonstrating qualitative and quantitative improvements in general texture synthesis performance. This suggests the generality of our formulation and demonstrates its potential for use in other applications of dynamic textures.

2. Background and Previous Work

Dynamic textures provide a rich stochastic model of image sequence appearance and dynamics. The classic dynamic texture model [6, 8, 20] has two parts. The first creates a low-dimensional representational model for the image appearance, often by computing the PCA decomposition of the set of images. The second fits an auto-regressive dynamic model predicting the coefficients of one image from the coefficients of the previous frames. Most recent uses of dynamic textures have had the goal of segmenting scenes into regions [3, 7, 23] or layers with specific dynamics patterns [4], or classifying videos based on the dynamics of the scene [2, 18, 19, 24, 9].

Approaches to improving the synthesis of new images have addressed both the limitations of a linear appearance basis and the limitations of a first order dynamics model. A manifold learning approach–followed by learning the dynamics of manifold coordinates–was used to characterize the space of appearance with more fidelity than linear models allow [13]. A second approach used a higher-order SVD (tensor decomposition) of the video to capture both spatial and temporal dynamics [5]. However, these approaches are more complicated and do not scale well to large datasets.

The image synthesis process of dynamic textures can be naturally driven through Kalman filtering. This has been used in the context of surveillance [25], where the entire (possibly corrupted or occluded) image is used as a measurement. Similarly, our approach uses a sparse set of noisy samples from an image in addition to the first use of implicit *a priori* smoothness constraints within a Kalman filter.

Computer vision techniques have been employed successfully for weather classification. For example, [14] addressed a two-class weather classification problem for arbitrary, non-stationary imagery using a manually-labeled training dataset. The application of capturing continental-scale cloud maps based on widely-distributed ground measurements was first considered in [11] by learning a linear regression estimator for the cloud cover map from PCA coefficients at each ground camera. Better approaches to learning the mapping between ground level imagery and local cloud estimates were presented in [16]. However, this paper is the first attempt to exploit the consistent patterns of how clouds propagate over time.

3. Dynamic Textures

The higher-order dynamic texture model described in [10] uses a linear dynamical system to predict future images. Each frame of the training sequence is mean-subtracted and projected onto a lower dimensional basis. Future coefficients are computed as a linear transformation of the coefficients of m previous frames. This process is summarized in Equation 1 where $y(t)$ are the images formed as column vectors, $\bar{y}$ is the mean image, and $\mathbf{C}$ is the reduced dimensionality basis. The state of the system is represented by the coefficients used to reconstruct an image, $x(t)$, and $\mathbf{A}_j$ are the state transition matrices. The state prediction $x(t)$ has a stochastic error w, assumed to be drawn from a Gaussian distribution with covariance $\mathbf{Q}$:

$$x(t) = \sum_{j=1}^{m} \mathbf{A}_j x(t-j) + w, \quad w \sim \mathcal{N}(0, \mathbf{Q}) \qquad (1)$$
$$y(t) \approx \bar{y} + \mathbf{C}x(t).$$

3.1. Hierarchical Extension

For many scenes, the challenge to creating high-fidelity images $y(t)$ is in the necessity to have many components in the linear decomposition, leading to a large coefficient vector $x(t)$. This increases the size of the $\mathbf{A}_j$ matrices characterizing system dynamics, and therefore increases the amount of necessary training data. Solving for a dynamic texture model explicitly within a multi-resolution structure allows for the dynamics and appearance model to share low-frequency components across larger areas but retain independent local models for the higher-frequency components.

To accomplish this, our generic formulation defines r regions in the image. We introduce a separate linear dynamical system for each of r arbitrarily partitioned regions so that each one (indexed by k) has a linear basis C_k for reconstructing its appearance, and its own independent state transition matrices $A^{k,l}$. These state transition matrices are constructed so that the coefficients of region k can be predicted from the history of the coefficients of some subset R_k of the regions (which could be, for example, the region itself and ancestor regions, as explained in detail later). Each region also has an uncertainty in its state transitions, represented by the covariance matrix Q_k.

The overall reconstructed image can be formed by taking the sum of each region's reconstruction from the coefficients $x_k(t)$. Equation 2 gives the model for predicting $x_k(t)$, the coefficients of region k at time t, as a linear transformation of the coefficients from the previous m frames of image regions in R_k. Images are reconstructed as the sum of the average image and the reconstructions of each region. This model reduces to that of ordinary dynamic textures if each region covers the entire image and each subset R_k contains all of the r regions.

$$x_k(t) = \sum_{l \in R_k} \sum_{j=1}^{m} \mathbf{A}_j^{k,l} x_l(t-j) + w_k, \quad w_k \sim \mathcal{N}(0, \mathbf{Q}_k)$$
$$y(t) \approx \bar{y} + \sum_{k=1}^{r} \mathbf{C}_k x_k(t) \qquad (2)$$

While the optimal region interactions and support are likely scene-dependent, in this work we consider a general quad-tree hierarchical structure. We find this to have sufficient expressibility to represent the appearance of many dynamic textures with improved performance, especially when combined with the smoothness constraints described in Section 4.1. This model restricts the complexity of the interactions between coefficients and can be interpreted as a regularization on the system parameters that allows for independent dynamics and makes it possible to solve with fewer training data.

Our quad-tree hierarchy starts at the top level with a single region covering the entire image. On the next level, we divide that region into quadrants and repeat this process recursively until the desired number of layers is reached. Each region has its own linear basis $\mathbf{C}_k$ that can be used to reconstruct the residual error left by its parent region, so the resulting image decomposition has many total variables. However, we model the dynamics of a region k as depending only on the coefficients of itself and its ancestors. Thus, in our generic formulation, we define R_k to contain the regions that are supersets of region k, which enforces that the dynamics of a region be defined by its own history and the history of its parent regions in the quad-tree hierarchy.

This design choice embodies the observation that lower-frequency image motion is often correlated over large image regions while high-frequency motion is correlated over small spatial scales. The spatial smoothness constraints in Section 4.1 address a side effect of this approach which can result in inconsistencies across region boundaries.

3.2. System Identification

Given sufficient training data, it is possible to solve for the matrices $\mathbf{A}_j^{k,l}$, $\mathbf{Q}_k$, and $\mathbf{C}_k$. While optimal solutions exist [22], they are infeasible for systems with a large number of variables and closed-form suboptimal solutions have been found to be very effective [20].

We follow the lead of the original dynamic textures model described in [20] and use a Principal Component Analysis (PCA) as the appearance basis in image regions. In our quad-tree decomposition, we borrow from an approach

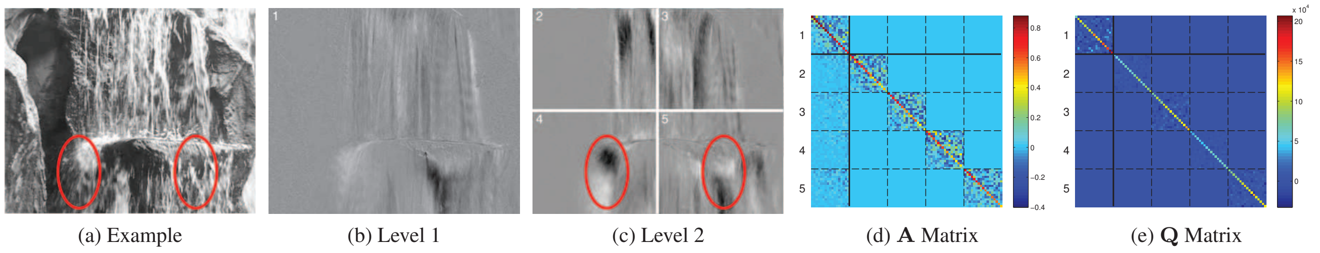

(a) Example (b) Level 1 (c) Level 2 (d) **A** Matrix (e) **Q** Matrix

Figure 2: An example of full, concatenated hierarchical dynamic textures system parameters. The sequence (a) demonstrates motion that is largely independent on the left and right side of the waterfall. The first level of the appearance decomposition (b) accounts for global pixel correlations, while the second level (c) decomposes each image quadrant separately. The state transition matrix (d) and the uncertainty covariance (e) have block-structured sparsity that enforces the independence of these higher level regions.

to hierarchical, hybrid PCA, which has been used in image denoising [15]. In its original form, this considers all patches in an image and computes a low-dimensional PCA reconstruction followed by computation of the residuals in each patch. Then, in each image quadrant, a PCA basis is constructed on the residuals of the patches in that quadrant. This process is repeated over several layers. For image denoising, this structure captures the fact that image textures (e.g. hair) are often localized to one part of the image, and a PCA patch basis can be constructed to more efficiently account for the residual error in regions that have consistent texture. In our context, these localized regions allow for the decoupling of distant, unrelated motions. Similarly, we create an appearance basis by first computing PCA over the entire image and calculating the reconstruction residuals. We then divide the image into quadrants and construct a PCA basis of the residuals in these smaller regions, repeating this process over several layers. A demonstration of this is shown in Figure 2(b-c).

Given this fixed basis, the remaining parameters can be learned using ordinary autoregressive methods. For first-order models that depend only on the coefficients from the previous time step, simple least-squares works well [20]. For higher-order models, insufficient training data and noise can cause the resulting dynamical system to be unstable. While many solutions have been proposed to resolve this issue [1], we had most success with the algorithm in [21], which estimates higher-order statistics recursively instead of computing them directly from training data.

Equation 2 can be reformulated by concatenating the coefficients for each region $x_k(t)$ into the set of all coefficients $x(t)$, combining the various prediction matrices $\mathbf{A}_j^{k,l}$ to create a single unified system of the form:

$$\begin{aligned} x(t) &= \mathbf{A}x(t-1) + w, \quad w \sim \mathcal{N}(0, \mathbf{Q}) \\ y(t) &\approx \mathbf{C}x(t) \end{aligned} \tag{3}$$

Since this is exactly the same form as ordinary dynamic textures, our approach can be interpreted as a method for enforcing predefined sparsity constraints on the parameters, as shown in Figure 2(d-e), which results in a system with improved prediction performance in comparison to ordinary dynamic textures with the same number of coefficients.

4. Integrating Image Measurements

Image synthesis using dynamic textures is often driven by random noise, but there are applications where some limited image information is available. For example, it may be possible to acquire a sparse set of pixels of the desired output image. In the case of constructing large-scale cloud maps, this information is obtained from localized ground-based webcam measurements. We would thus like to incorporate these measurements within the synthesis process. Since dynamic textures–including our hierarchical version–are expressed in terms of a linear state update and a linear mapping between the state and the image, a Kalman filter provides a natural means to do so.

Linear dynamical systems, as in Equation 3, can be optimally filtered when observations of the true state are available. Given measurements (possibly corrupted by Gaussian noise), state update models, and appropriate estimates of the noise distributions, the Kalman Filter gives this optimal fusion of measurements with model dynamics[12].

In standard terminology, $x(t)$ is the state vector at time t, $\mathbf{A}$ is the state transition model that predicts the state vector from its previous value, w is the process noise, $z(t)$ is a vector of measurements at time t, and $\mathbf{H}$ is the linear observation model that maps $x(t)$ to $z(t)$. The measurement noise v is assumed to be normally distributed with a covariance $\mathbf{R}$ estimated from training data. The measurements then can be expressed as:

$$z(t) = \mathbf{H}x(t) + v, \quad v \sim \mathcal{N}(0, \mathbf{R})$$

During the prediction step, the Kalman filter estimates the *a priori* state prediction and updates the uncertainty accordingly, as shown in Equation 4 below.

$$\hat{x}(t) = \mathbf{A}x(t-1), \quad \hat{\mathbf{P}}_t = \mathbf{A}\mathbf{P}_{t-1}\mathbf{A}^\intercal + \mathbf{Q} \tag{4}$$

The optimal Kalman gain $\mathbf{K}_t$ is then used in the update step to compute the *a posteriori* estimates with respect to the observation $z(t)$, as shown in Equation 5.

$$\mathbf{K}_t = \hat{\mathbf{P}}_t \mathbf{H}^\intercal \left(\mathbf{H} \hat{\mathbf{P}}_t \mathbf{H}^\intercal + \mathbf{R} \right)^{-1}, \quad \mathbf{P}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}) \hat{\mathbf{P}}_t$$
$$x(t) = \hat{x}(t) + \mathbf{K}_t (z(t) - \mathbf{H}\hat{x}(t)) \qquad (5)$$

Since measurements are linear transformations of the state vector, they can be expressed directly in image space as linear combinations of pixel values by first projecting the state coefficients onto the basis $\mathbf{C}$. Thus, the measurement observation model can be written as $\mathbf{H}_m = \mathbf{S}_m \mathbf{C}$, where $\mathbf{C}$ is the basis and each image measurement in $z(t)$ corresponds to a row of the measurement pixel selection matrix $\mathbf{S}_m$, which contains the weights of the pixel values that make up the measurement. For example, if a measurement is sampled from a single pixel in the image, the corresponding row in $\mathbf{S}$ contains a value of one at the sample location and zeros elsewhere, selecting just the image pixel at the desired location. However, since a measurement can be constructed as a linear combination of any number of pixels, more complex measurements can also be represented.

4.1. Imposing Image Constraints

The hierarchical dynamic texture model has no explicit way of enforcing smoothness across boundaries between independent regions. However, because image measurements can be any linear combination of pixel values, they can be used to express priors on the expected appearance of the synthesized images. To constrain two neighboring pixels to be equal, the row in a constraint pixel selection matrix $\mathbf{S}_c$ will contain a value of one at the the first sample location and a value of negative one at the second sample location with zeros elsewhere. Thus, we can concatenate one (or many) "measurements" in $z(t)$ that are fixed to be zero for a set of constraints $\mathbf{H}_c = \mathbf{S}_c\mathbf{C}$ defined as the difference of neighboring pixels across region boundaries, where the sum of each row in $\mathbf{S}_c$ is constrained to be zero.

5. Experimental Results

In this section, we describe our experiments in the tasks of cloud map synthesis from webcam imagery and general dynamic texture synthesis. We use RMS pixel error as a comparison metric for all of our experiments. While this is generally not appropriate for evaluating qualitative image synthesis, we emphasize that our primary goal is not to produce visually similar images (like traditional dynamic textures) but to estimate physically-meaningful quantities (e.g. IR reflectance, which is related to cloud elevation and thickness) arranged on a grid.

5.1. Cloud Map Synthesis

We evaluate the applicability of hierarchical dynamic textures in the real-world application of constructing large-scale maps of cloud cover from sparse measurements acquired from distributed, ground-based webcams. Specifically, we train regression models on one year of imagery from 4388 webcams across the United States to predict scalar measurements of local cloudiness. We adopt the approach of [16] and first decompose the imagery using partial least squares, retaining the top 20 components that best predict cloudiness sampled from historical satellite imagery. We then train random forest regression models using this data with 200 trees and all 20 features. Since many of the webcams were concentrated close together in highly-populated areas, we include only the measurement with the lowest prediction error variance (estimated on validation data) in a radius of 10 pixels in the satellite image, or approximately 100 miles. In addition, we train separate models for both day and night images, thus taking advantage of cameras in which clouds are visible at night due to city lights, moonlight, etc.

A first-order hierarchical dynamic texture model, as described in the previous sections, was trained on 3000 satellite images using 4 levels with 5 principal components per region for a total of 425 variables. In addition, we included 300 Gaussian filtered consistency constraints uniformly distributed across region boundaries with variances estimated from training data. Figure 3 shows the average numerical prediction results of a model trained on 6 months of data. Despite the large amount of noise in the cloudiness predictions, our model performs better than ordinary dynamic textures and natural neighbor spatial interpolation, which showed the best performance in a thorough comparison of cloud map interpolation methods [16]. In addition, by considering ideal measurements sampled directly from ground truth satellite imagery, we show that our model has significant room for improvement if more accurate regression could be available. Finally, Figure 6 shows example synthesized cloud maps. (Videos are also included in the supplementary material to accentuate the temporal consistency gained by considering cloud dynamics.) While the output of the ordinary dynamic textures model appears sharper and more perceptually similar to the ground truth in some cases, this hallucinated high-frequency content is simply overfitting due to the measurement sparsity and the large spatial extent of the output, in which a pixels width is on the order of 10km. This is verified quantitatively in Figure 3 which clearly shows improved prediction performance of our model.

5.2. General Experiments

We also quantitatively and qualitatively compare the performance of hierarchical and ordinary dynamic textures on

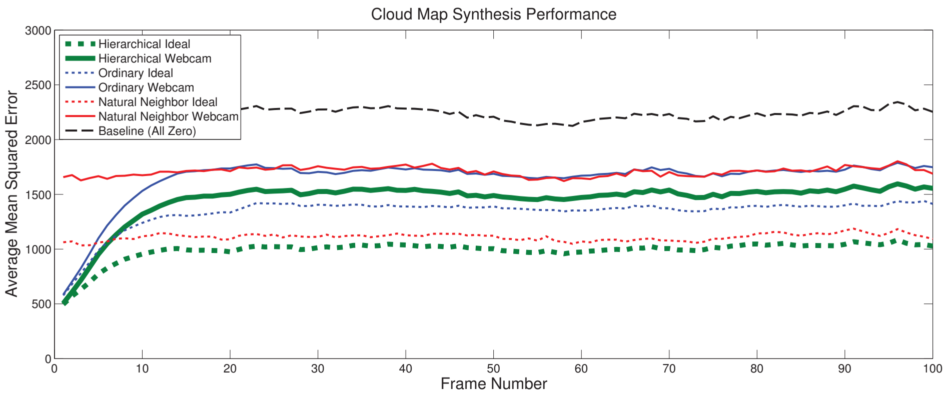


Figure 3: Satellite image synthesis performance in groups of 100 frames averaged over six months of testing data. Green lines show Kalman filtering of our proposed hierarchical dynamic texture formulation, blue lines show ordinary dynamic textures, and red lines show natural neighbor spatial interpolation. Solid lines use actual noisy webcam measurements while dashed lines use ideal (simulated) measurements obtained by sampling the ground truth satellite image at the same locations. The true satellite images prior to frame 1 are known exactly for initialization and the Kalman filters converge after approximately 20 frames. Our method almost always results in lower error both with ideal and webcam-derived measurements.

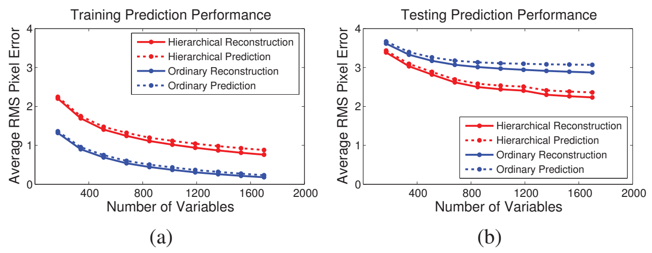


Figure 4: The reconstruction and single-step prediction error for both training (a) and testing (b) data in response to increasing the number of variables of ordinary and three-level hierarchical dynamic textures. The systems were trained on a sequence of 3000 satellite images and example frames are shown on the right. The ordinary dynamic texture model with the same number of total coefficients performs better on training data while the proposed model generalizes significantly better to testing data.

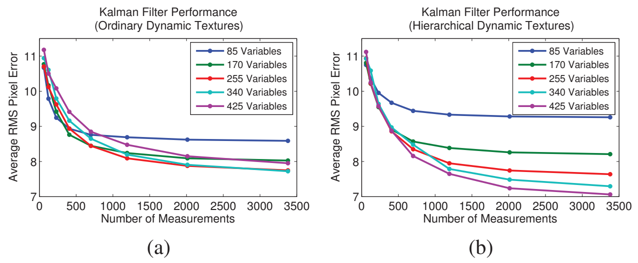


Figure 5: A comparison of Kalman filtering performance using both ordinary (a) and hierarchical (b) dynamic textures while varying the total number of variables and the number of observed sparse pixel measurements. While ordinary dynamic textures result in better performance with fewer coefficients, a hierarchical model can more effectively integrate many measurements for more accurate reconstructions of localized details, as shown by the decreased error in models with more variables.

a variety of sequences from the DynTex database [17]. Figure 4 demonstrates the improved generalizability of the hierarchical approach (red lines) in comparison to ordinary dynamic textures (blue lines), which is based on an appearance model built from PCA over the entire spatial extent of the image. For a given set of coefficients, PCA is guaranteed to optimally reconstruct the training data, which leads to better reconstruction and prediction performance on that data simply because the reconstructions are better (Figure 4a). On the other hand, the hierarchical model gives improved performance on testing data with the same number of total coefficients indicating reduced overfitting (Figure 4b). This likely indicates that our model better generalizes to novel data during testing because it is able to consider smaller regions independently.

Figure 7 shows a similar comparison across 24 different sequences from the DynTex database. These sequences were all tested with a 500 image training sequence and a hierarchical model consisting of 3 layers with 10 components per region for a total of 210 coefficients, which we compare to ordinary Dynamic Textures using 210 basis images computed through PCA. We plot the difference in the reconstruction error, comparing the hierarchical model to the ordinary Dynamic Textures model. While our approach results in greater training reconstruction and prediction errors, there is almost always improvement in reconstructing and predicting testing data (which begin 20 frames after the end of the training data) indicating better generalizeability.

The hierarchical model also allows for more accurate reconstructions when sparse measurements are available. Figure 5 demonstrates this by considering a four-level hierarchical model in comparison to ordinary dynamic textures with the same number of variables. Both were trained on 582 frames at the beginning of the sequences and evaluated

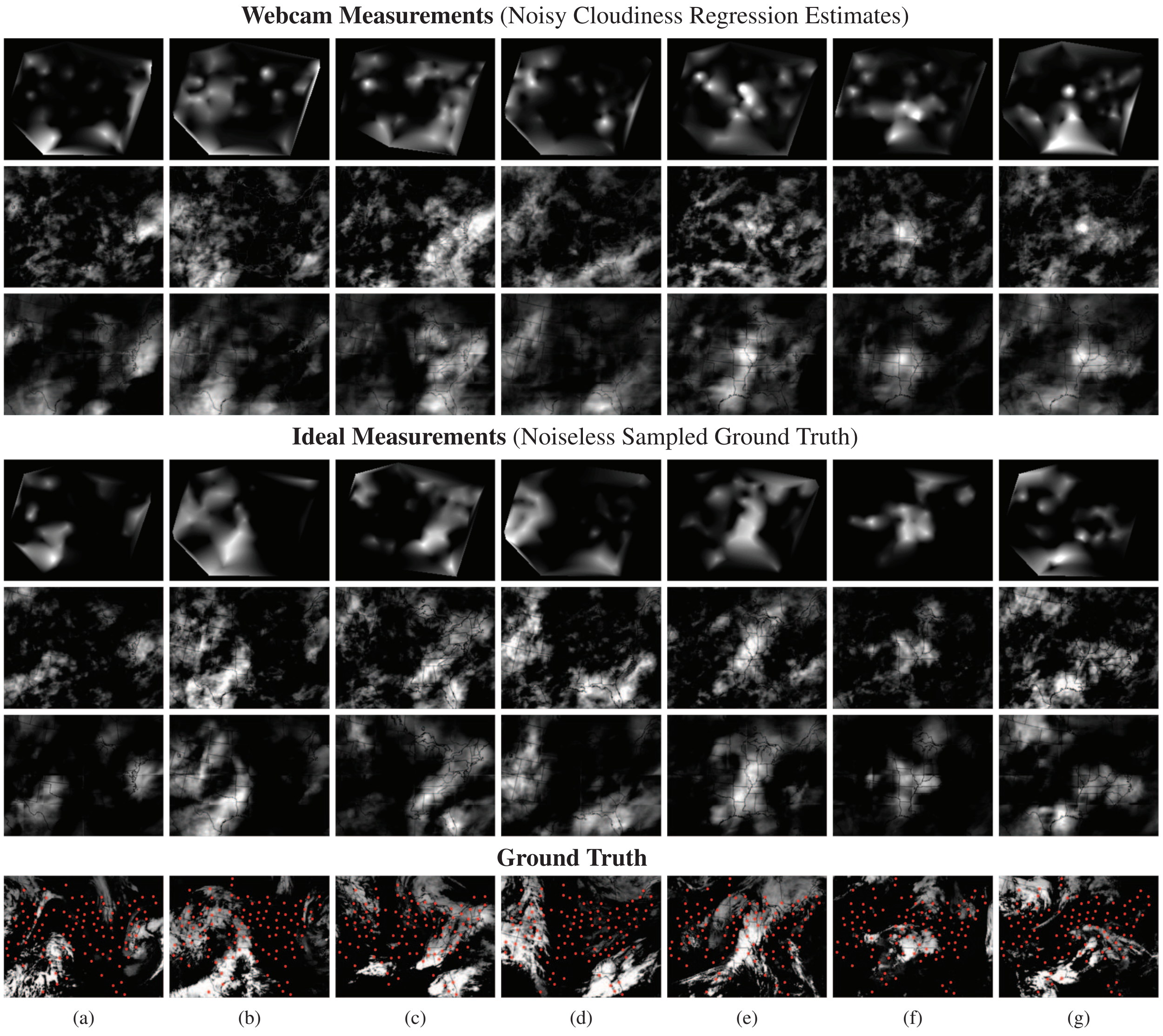

Webcam Measurements (Noisy Cloudiness Regression Estimates)
Ideal Measurements (Noiseless Sampled Ground Truth)
Ground Truth
(a)
(b)
(c)
(d)
(e)
(f)
(g)

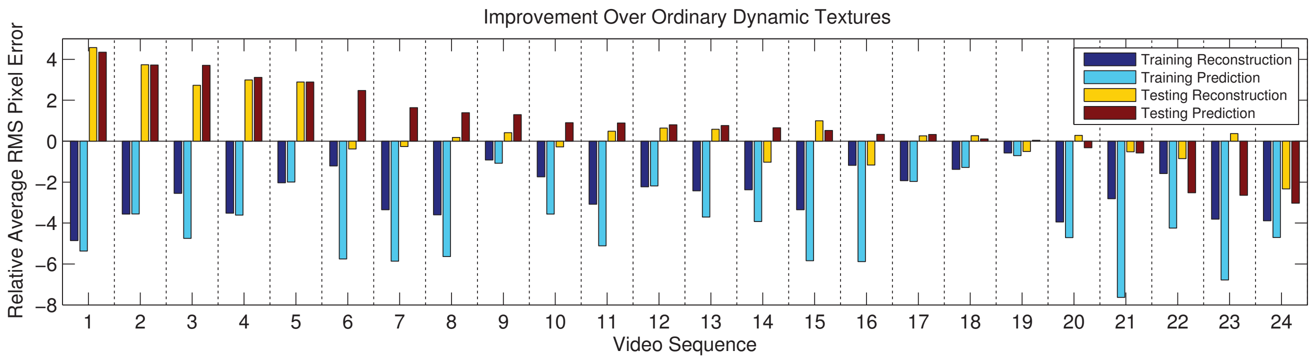
Figure 6: Synthesis results (a-g) for example satellite imagery. (Full video sequences are found in the supplementary material.) Rows 1-3: Frames synthesized from noisy webcam measurements using nearest neighbor interpolation, ordinary dynamic textures, and hierarchical dynamic textures with edge consistency constraints respectively. (All frames were taken sufficiently far, i.e. over 50 frames, from initialization.) Rows 4-6: Frames synthesized from ideal measurements sampled from the ground truth using the same methodology. Note that our proposed approach (Rows 3,6) results in images that are less noisy and more robust to measurement error. Row 1: The ground truth satellite image with webcam measurement locations shown as red dots.

on a test set of 282 frames beginning 20 frames after the end of the training data. The hierarchical model is better able to accommodate many variables resulting in reduced error.

6. Conclusions

We demonstrated that available data can give insight for tailoring existing methods (e.g. dynamic textures) to new applications (e.g. meteorology.) Specifically, for the task of constructing large-scale maps of cloud cover from ground-based webcam imagery, the multi-scale behavior of clouds inspired our hierarchical model and the public availability of geolocated webcams provided a means to drive its dynamics using sparse measurements within a Kalman filter. We believe this fusion of seemingly disparate data sources will yield even better results as online image archives become even more ubiquitous.

In comparison to previous approaches, our method was able to train more representative dynamic models using a novel representation that decoupled distant, independent regions of cloud cover while enforcing consistency between them. In addition to providing improved interpolation performance of cloudiness measurements, we demonstrated potential for our model to be applied to other dynamic texture applications due to its better generalizability.



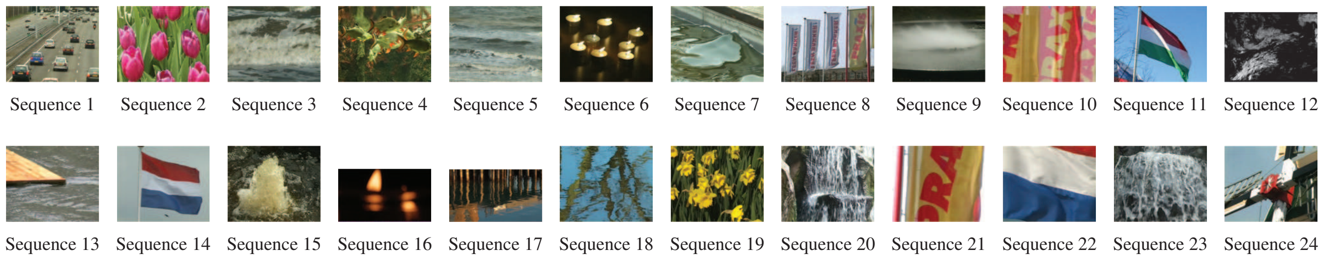


Figure 7: Relative performance for 24 sequences from DynTex. The y-axis shows difference of reconstruction error of proposed hierarchical model relative to ordinary dynamic textures, where positive values show an improvement for the proposed model. Reconstruction and single-step prediction errors are shown for both training and testing data. Ordinary dynamic textures always performs better on training data, but hierarchical extension usually improves results on the testing data indicating reduced overfitting and better generalizability. The biggest improvement is in sequences that exhibit complicated, locally-correlated motion like cars (Sequence 1), while standard dynamic textures do better with globally-correlated, deterministic motions like windmills (Sequence 24).

Acknowledgments: We gratefully acknowledge the support of our sponsors. Calvin Murdock was partially supported by the Department of Defense (DoD) through the National Defense Science and Engineering Graduate Fellowship (NDSEG) Program, Nathan Jacobs by DARPA grant D11AP00255, and Robert Pless by NSF grants DEB-1053554, EF-1065734, and NSF IIS-1111398.

References

[1] B. Boots, G. J. Gordon, and S. M. Siddiqi. A constraint generation approach to learning stable linear dynamical systems. In *Advances in Neural Information Processing Systems*, 2008. 4

[2] A. Chan and N. Vasconcelos. Classifying video with kernel dynamic textures. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 2

[3] A. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):909–926, 2008. 2

[4] A. Chan, N. Vasconcelos, et al. Layered dynamic textures. In *Advances in Neural Information Processing Systems*, volume 18, page 203, 2006. 2

[5] R. Costantini, L. Sbaiz, and S. Susstrunk. Higher order svd analysis for dynamic texture synthesis. *IEEE Transactions on Image Processing*, 17(1):42–52, 2008. 2

[6] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003. 2

[7] G. Doretto, D. Cremers, P. Favaro, and S. Soatto. Dynamic texture segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2003. 2

[8] A. Fitzgibbon. Stochastic rigidity: Image registration for nowhere-static scenes. In *Proc. IEEE International Conference on Computer Vision*, 2001. 2

[9] K. Fujita and S. Nayar. Recognition of dynamic textures using impulse responses of state variables. In *Proc. International Workshop on Texture Analysis and Synthesis (Texture 2003)*, 2003. 2

[10] M. Hyndman, A. Jepson, and D. Fleet. Higher-order autoregressive models for dynamic textures. In *British Machine Vision Conference*, 2007. 3

[11] N. Jacobs, S. Satkin, N. Roman, R. Speyer, and R. Pless. Geolocating static cameras. In *Proc. IEEE International Conference on Computer Vision*, Oct. 2007. 2

[12] E. Kalman, Rudolph. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960. 4

[13] C. Liu, R. Lin, N. Ahuja, and M. Yang. Dynamic textures synthesis as nonlinear manifold learning and traversing. In *British Machine Vision Conference*, 2006. 2

[14] C. Lu, D. Lin, J. Jia, and C.-K. Tang. Two-class weather classification. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 2

[15] J. Mairal, G. Sapiro, and M. Elad. Learning multiscale sparse representations for image and video restoration. *SIAM Multiscale Modeling and Simulation*, 7(1):214–241, 2008. 4

[16] C. Murdock, N. Jacobs, and R. Pless. Webcam2satellite: Estimating cloud maps from webcam imagery. In *Proc. IEEE Workshop on Applications of Computer Vision (WACV)*, 2013. 1, 2, 3, 5

[17] R. Péteri, S. Fazekas, and M. J. Huiskes. DynTex : a Comprehensive Database of Dynamic Textures. *Pattern Recognition Letters*, 31(12):1627–1632, 2010. http://projects.cwi.nl/dyntex/. 2, 6

[18] A. Ravichandran, R. Chaudhry, and R. Vidal. View-invariant dynamic texture recognition using a bag of dynamical systems. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 2

[19] P. Saisan, G. Doretto, Y. Wu, and S. Soatto. Dynamic texture recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2001. 2

[20] S. Soatto, G. Doretto, and Y. N. Wu. Dynamic textures. In *Proc. IEEE International Conference on Computer Vision*, 2001. 2, 3, 4

[21] O. Strand. Multichannel complex maximum entropy (autoregressive) spectral analysis. *IEEE Transactions on Automatic Control*, 22(4):634–640, 1977. 4

[22] P. Van Overschee and B. De Moor. N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1):75–93, 1994. 3

[23] R. Vidal and A. Ravichandran. Optical flow estimation & segmentation of multiple moving dynamic textures. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005. 2

[24] F. Woolfe and A. Fitzgibbon. Shift-invariant dynamic texture recognition. In *Proc. European Conference on Computer Vision*. Springer, 2006. 2

[25] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *Proc. European Conference on Computer Vision*, 2003. 2