Environment Prediction from Sparse Samples for Robotic Information Gathering

Jeffrey A. Caley[1] and Geoffrey A. Hollinger[2]

Abstract— **Robots often require a model of their environment to make informed decisions. In unknown environments, the ability to infer the value of a data field from a limited number of samples is essential to many robotics applications. In this work, we propose a neural network architecture to model these spatially correlated data fields based on a limited number of spatially continuous samples. Additionally, we provide a method based on biased loss functions to suggest future areas of exploration to minimize reconstruction error. We run simulated robotic information gathering trials on both the MNIST hand written digits dataset and a Regional Ocean Modeling System (ROMS) ocean dataset for ocean monitoring. Our method outperforms Gaussian process regression in both environments for modeling the data field and action selection.**

I. Introduction

Robots often require a model of their environment to make informed decisions. In unknown environments, the ability to infer the value of a data field from a limited number of samples is essential to many robotics applications. A robot in an unknown building capable of estimating the building layout will be able to explore and search much safer. A self-driving car can travel much faster and more safely if it can anticipate future events by observing the past. An ocean glider can more efficiently navigate through current fields with estimates of ocean current magnitudes. Efficient methods for modeling and determining what data is sampled are essential to the quality of the inference and the performance of the robot.

A number of previous estimation methods use Gaussian Process (GP) regression to approximate environmental conditions. While GPs provide model-free estimates with confidence bounds that can be leveraged to select future samples, these samples need to be geographically diverse for the GP to provide an accurate estimate over the entire data field. In applications where geographies are large and travel times are long, collecting a diverse set of samples is often not feasible. A robot traveling through such an environment will collect a series of collocated samples which will result in accurate estimations near the samples, but poor estimation far away. To address this issue, we present a new modeling technique for spatially correlated data fields.

This paper describes a collection of novel encoder-decoder deep neural networks that exploit historical data to model data fields from a limited number of continuous observations. We leverage a network architecture used for image inpainting, the task of filling in holes in an image, to infer the data field from a limited number of samples. The network is trained on historic data and example sampling paths. To facilitate the selection of future samples, our method trains two additional networks to serve as arbiters of information value. One network is trained to err on the side of overestimation, while the other is trained to underestimate. These two networks can then be used as a measure of variance to investigate future samples for potential information gain.

We demonstrate the benefit of our approach by reconstructing both static and time series environments taken from both the MNIST dataset and ocean model data from a Regional Ocean Modeling System (ROMS) dataset. In a series of simulated robotic information gathering tasks, we compare the estimation accuracy of a GP to the proposed approach using both random sampling of observation locations and variance minimizing sampling. We show the proposed technique outperforms the state-of-the-art.

II. Related Work

A. Robotic Information Gathering

Robotic information gathering problems involve modeling an unknown environment or phenomenon limited by a discrete number of samples. Modeling entails inferring the value of unmeasured data points while sampling refers to taking the measurement of a data field at a specific point. Both modeling and sampling strategies come with challenges. Techniques based on probabilistic models, such as Gaussian Processes, are common due to their ability to provide uncertainties along side their estimate. These uncertainties can then be leveraged to define sampling strategies, such as sampling to minimize uncertainty [1] or entropy [2]. It has been shown that the sequential greedy observations performs well in submodular cases [3].

Particularly relevant to this work are information-theoretic approaches. Bayesian optimization is used in [4] to predict mutual information over a series of candidate actions, analogous to our proposed method. However in this case, a single action results in a wide range of samples used to train a GP. Sampling techniques based on Upper Confidence Bound (UCB) are performed in [5] to find the highest value sample. This technique requires a variable of interest and does not lend itself directly to exploration. Other work focused on view planning has had a direct influence on our work [6]. These authors train a deep network to take an image of an object from a single pose and use it to reconstruct a

*This work was funded in part by Office of Naval Research grant N00014-17-1-2581

[1]Computer Science Department, Pacific Lutheran University, Tacoma, Washington 98444 - caleyjb@plu.edu

[1] Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University, Corvallis, Oregon 97331 - geoff.hollinger@oregonstate.edu

series of images that represent the object in its totality. The network also suggests the next viewing angle of the object to minimize the reconstruction error in future estimates. While producing promising results, their network architecture limits the network's ability to predict images that are different from the inputs. We look to address these problems through our proposed architecture.

B. Environment Modeling

While picking the correct place to sample is important, it is meaningless without an effective model to utilize these samples to infer the missing information. Models based methods, such as the publicly available Regional Ocean Modeling System (ROMS) [7], uses sophisticated models of physical ocean processes combined with forward simulations of measured data to estimate ocean information such as currents, temperatures, salinity, etc.

By treating data fields as images, image processing approaches can be applied. Specifically, a subset of image processing research called image inpainting focuses on filling in holes missing from the image. Taken to the extreme, where the large majority of the image is missing, it is analogous to modeling a data field with limited samples. Non-learning approaches to image inpainting rely on propagating pixel information from nearby to the missing area [8]. This method only works when the missing area is very small. Patch-based methods look for areas in the non-hole image that may match stylistically, but this also presents problems if the missing data is unique to that image [9], [10].

Deep learning image inpainting methods often use a placeholder to identify the missing region and pass it through a convolutional network to learn the missing data [11]. Generative adversarial network (GAN) [12] have inspired solutions formulated as a conditional image generation problem where high and low-level information is synthesized to generate missing information through an adversarial approach [13], [14], [15]. The U-net architecture [16], often used in image segmentation problems, is used in [17] in conjunction with a partial convolution layer to fill irregularly shaped holes in images. While these works are promising for image reconstruction, their focus on photo realistic inpainting is not necessary for environment modeling. These techniques also do not provide any insight into where future information should be gathered to further reduce errors. The work proposed in this paper combines the advantages of deep learning with the body of knowledge built up in the robotic information gathering community around GPs. We harness the power to learn from large datasets while developing a method to guide future data collection.

III. Problem Formulation

In this work we consider the problems of predicting the value of a two- or three-dimensional environment given a limited number of samples. Starting from a random location within the field, a robot is tasked with moving within the environment to collect data for the purpose of reconstructing the most accurate estimate of the environment. The goal of the robot is to minimize the error between its estimate and the true value.

More formally, we assume a bounded planar workspace $\Omega \subseteq \mathbb{R}^2$ or $\mathbb{R}^3$. This workspace is initially unknown to the exploring robot. During exploration along a trajectory $\mathcal{P}$, selected from the space of all trajectories, Ψ, the sensors on the robot reveal the region $\omega_{known}(\mathcal{P})$. The optimal path $\mathcal{P}^*$ is the path that produces the most accurate reconstruction of Ω given a prediction model $I(\omega_{known}(\mathcal{P}))$ and some budget B (e.g., time, fuel, or energy). Thus, we wish to solve the following optimization problem:

$$\mathcal{P}^* = \underset{\mathcal{P}\in\Psi}{\operatorname{argmax}} \; I(\omega_{known}(\mathcal{P})) \quad \text{s.t. } c(\mathcal{P}) < B. \tag{1}$$

IV. Method

The proposed method can be divided into two portions: an offline section for training networks and an online portion where the networks are utilized to estimate data fields and select future actions. The offline portion consists of training three neural networks on a training set representative of the domain being modeled. An optimistic network is trained to estimate the data field, but to err on the side of overestimating the data. A pessimistic network is trained to err on the side of underestimating the data. The non-biased network is trained to estimate the field with no bias.

The networks are trained by randomly generating sample robot trajectories. These trajectories are provided as masks to the neural network. The masks are binary matrices which indicate which areas of the environment have been sampled and which have not. The network is trained using supervised learning to predict the full data field given an input which has been masked off to only provide sampled data to the neural network.

Once the networks are trained, we can leverage them to model and explore the data field in question. During online operations, the non-biased network provides a best estimate of the data field given the current samples. To evaluate potential future actions, every potential next action is evaluated for its information potential. Potential information is estimated by 'hallucinating' the collected samples given an action using the current best estimate of the data field as ground truth. These samples constitute the inputs into the optimistic and pessimistic networks. The difference between the two network estimates provides a heuristic of uncertainty and act as a metric for information gained. The action that produces the smallest difference between the estimates provides the most information and is chosen as the next action. This process is repeated until the budget is exhausted and a final estimate of the environment can be generated.

A. Network Architecture and Implementation

The network architecture and design has been split into two categories: two-dimensional fields and three-dimensional time varying fields. The two-dimensional case assumes the underlying robotic environment is not changing with time, such as a robot exploring a building. The three-dimensional

case investigates robotic environments where the environment changes as the robot explores, such as ocean environments.

1) Network Design - 2D: We propose using a neural network motivated by the architecture presented in [17] for modeling data fields given a limited number of samples. As input, the network accepts a data field of sampled points and a binary mask indicating the locations of the sampled points. Through a process of convolving learned filters over sampled data, the network iteratively fills in the missing data. The architecture provides an avenue for local low level features to be applied to local areas, while also learning higher level global features that can be used across large areas of the environment. The network outputs an estimate of all values in the sampled data field. Figure 1 provides a detailed illustration of the architecture.

2) Network Design - 3D: The proposed network architecture to handle time varying data builds upon our two-dimensional design. The data in the three-dimensional case is time series data. To handle this, we propose a recurrent network utilizing convolutional LSTM layers with modified partial convolution to handle sparse data. The network has a similar architecture to the 2D case, with an encoder-decoder network architecture that iteratively fills in the missing data through partial convolution. The recurrent nature of the network provides a memory of previously sampled data, allowing the network to learn how the data field will change through time and use previously sampled data in its current estimate. Each encoder layer has a corresponding decoder layer, which receives the encoded information along with all high level encoder information. These two sources of information are concatenated together in the filter dimension and used as inputs into the current masked convolutional LSTM layer. This allows the network to utilize high level information from the encoder without losing lower level information from the less encoded pieces of data. The same process occurs for the LSTM memory as well. Figure 2 provides a detailed illustration of the network architecture.

The partial convolution LSTM (PCNN-LSTM) is a layer that combines a convolutional LSTM layer with partial convolution. This is done through a few steps. Using a convolutional LSTM layer as the base, the convolution is replaced with partial convolution from [17]. After each gate in the LSTM has applied its activation function, the mask is multiplied by the gate output. This zeros out any data outside the new unmasked region. Because the forget gate multiplies its output to the LSTM memory, and because the forget gate has masked off values, we need to avoid deleting sections of memory via the masked off values. To prevent this from happening, the forget gate multiplication only occurs in unmasked regions of the image. Following the formal definition of an LSTM layer, PCNN-LSTM is formally defined by equations 2 and 3:

$$\mathbf{f_t} = \sigma(\mathbf{W_f} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_f}) * \mathbf{mask}, \quad (2)$$

$$\mathbf{C_t} = (f_t * \mathbf{C_{t-1}} + (\mathbf{1} - \mathbf{mask}) * \mathbf{C_{t-1}}) + \mathbf{i_t} * \tilde{\mathbf{C}}_\mathbf{t}, \quad (3)$$

where $\mathbf{f_t}$ is the output gate and $\mathbf{C_t}$ is the output memory at time t. $\mathbf{mask}$ is a binary matrix indicating where samples have been taken.

B. Biased Networks and Action Selection

To facilitate the action selection process, we train three networks using the architecture described above. What differentiates these networks from each other is their loss functions. For the non-biased network, we use L1 loss. For the two biased networks, instead of using the L1 loss, we penalize any estimate that is either below (for the optimistic network) or above (for the pessimistic network) the true value. D_{below} and D_{above} represents a binary mask of points where the estimate was either below or above the ground truth. $I_{out} - I_{gt}$ is the ground truth of a current frame while I_{out} is the estimated field calculated by the neural network:

$$||_{1_{OptNN}} = |I_{out} - I_{gt}| + |(I_{out} - I_{gt}) * (D_{below} * 0.5)|, \quad (4)$$

$$||_{1_{PessNN}} = |I_{out} - I_{gt}| + |(I_{out} - I_{gt}) * (D_{above} * 0.5)|. \quad (5)$$

These loss functions push their respective network to either be conservative (producing images with lower values) or liberal (producing images with higher values) in their estimates. This provides a heuristic variance for our network estimates that can be utilized to determine which unsampled areas have high quantities of information.

Action selection is the process of selecting the next action to execute, a, from the set of all possible actions, $\mathcal{A}$, that maximizes our image reconstruction accuracy after new information has been sampled. This starts with our unbiased network, UnbiasedNN, predicting the value of the data field given its inputs. The estimated field, Est, serves as 'ground truth' as we hallucinate performing all available future actions and collecting a set of new approximated observations, $\hat{Obs}$. Each hallucination, $AppxObs()$, results in a larger set of sampled information that is used as inputs into our two biased networks, $OptNN$ and $PessNN$. These biased networks return two different estimates of the data field, Est^+ and Est^-. The L1 loss between these two fields is calculated and used as an information gain metric, R. If the two fields have a large L1 loss, it means the networks are unsure of their estimate, implying that the recently acquired samples did not provide quality information. A small L1 loss means the two networks have estimated a similar field despite their differences. This implies the samples collected provided quality information. Once each actions' L1 loss score has been calculated, the action with the smallest resulting L1 loss is estimated to have provided the most information to the network, thus being selected as the next action to take. This action is then executed, more observations are made and the unbiased network can make another estimate. This process repeats itself until the budget expires as shown in Alg. 1.

V. Experiments, Results and Discussion - 2D

To show that our proposed method generates quality data field estimates and produces informative actions, we compare our method to Gaussian Process regression in two different domains. The first domain uses the MNIST dataset [18].

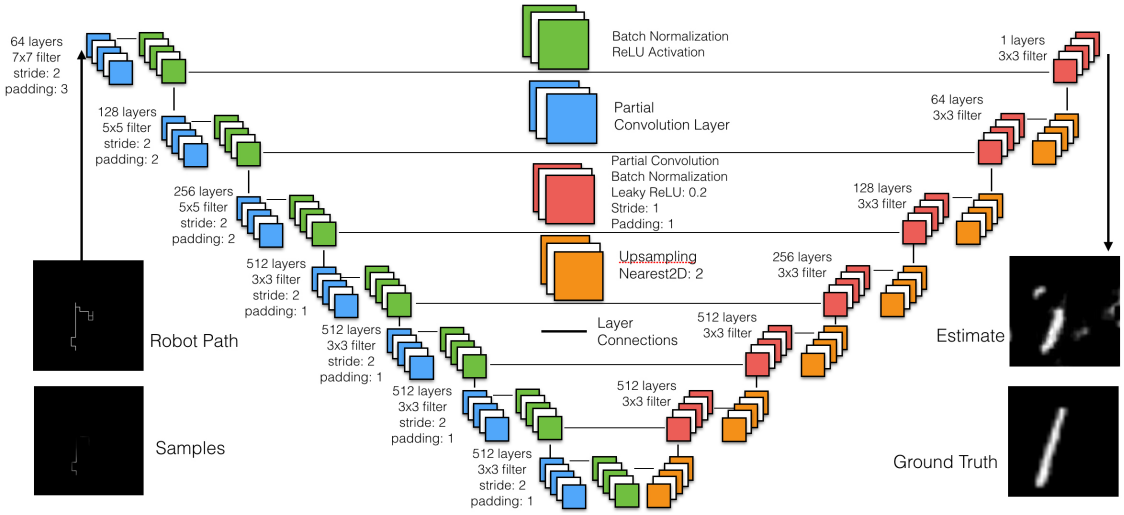

Fig. 1: An illustration of our network structure. The network takes as input a data field of sampled data points and a binary mask indicating which points in the data field are missing. The network performs a series of partial convolutions with stride 2, shrinking the image size before upsampling back to the original size. The network learns to predict the value of the missing points in the data field by learning over a training set of similar data fields and example binary masks.

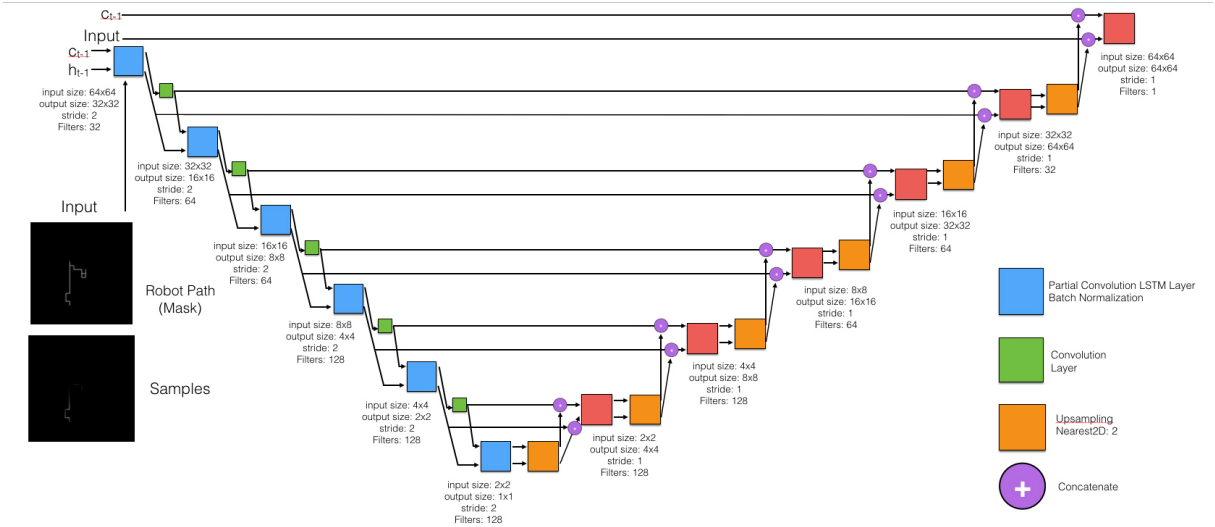

Fig. 2: An illustration of the network structure for PCNN-LSTM. The network takes as input a data field of sampled data points, a binary mask indicating which points in the data field are missing, a memory state and the output at $t-1$. The network performs a series of partial convolutions with LSTM at stride 2, shrinking the image size before upsampling back to the original size. The network learns to predict the $t+1$ value of the missing points in the data field by learning over a training set of similar time series data fields and example binary masks.

MNIST is a dataset of 50,000 handwritten digits, one through nine, prevalent in the image processing community. The second domain is a ROMS ocean dataset from Monterey Bay, CA [7]. The ROMS data provides a realistic data setting for which this method could be utilized for underwater vehicle monitoring tasks. In the two dimentional case, a single frame from the time series ROMS data is used.

To evaluate action selection, we compare our proposed bias network method to random actions, GP variance and CNN variances generated through dropout.

A. *MNIST*

We simulate a robot as it explores an environment represented by the MNIST dataset that has been upsampled to a 256x256 image using bilinear filtering. We train our networks on the training set and test on the test set provided by [19]. While MNIST seems like an unusual test domain for robotic information gathering problems, it provides some benefits over other datasets. With the MNIST data, it is apparent to a human observer where informative and non-informative samples are. Exploring areas on the digit are much more valuable then exploring the black regions of the

Algorithm 1 Exploration

procedure EXPLORATION($Obs, Budget$)
  if $Budget\ != 0$ **then**
    $Est =$ UnbiasedNN(Obs) // Calculate CNN estimate
    for $a \in \mathcal{A}$ **do**
      $\hat{Obs} =$ AppxObs(Est, a) $+ Obs$ // 'hallucinate' taking action a
      $Est^{+} =$ OptNN($\hat{Obs}$) // estimate bias1CNN output
      $Est^{-} =$ PessNN($\hat{Obs}$) // estimate bias2CNN output
      $R = ||Est^{+} - Est^{-}||_1$
    $a^* = \text{argmin}_{a \in \mathcal{A}}(R)$
    $newObs =$ Execute(a^*) // take action a*
    $Obs = Obs + newObs$ // collect new observations
    Exploration($Obs, Budget -$ Cost(a)) // continue exploration

image. With this in mind, we start a simulated robot with spatially localized sensing, limited to one pixel, in a random location somewhere on the digit. The robot starts by traveling 10 pixels up the image, sampling the value of all pixels traversed. With these initial samples, the robot calculates its next action and collects more samples during the chosen action. We limit the robot's actions to traveling in the four cardinal directions and have it traverse 10 pixel during each action. The robot is provided a budget of 25 to 100 actions, travels 10 pixels per action and takes actions randomly. For simplicity, if the robot tries to travel outside the image, we simply wrap the path to the other side of the image. The results of our CNN based approach compared to a GP can be seen in table I.

TABLE I: The average pixel error between estimated data field and ground truth over 100 MNIST images. The robot collects 10 samples per action and actions are taken at random. Budgets of 25, 50, 75, and 100 are shown. Error bars show one SEM.

	25	50	75	100 Actions
GP	$47.82 \pm .71$	$47.21 \pm .70$‘	$46.33 \pm .68$	$45.60 \pm .68$
CNN	36.06 ± 1.02	$32.18 \pm .96$	$29.23 \pm .98$	$26.87 \pm .98$

Evaluation of action selection is displayed in Table II. All values are expressed as a proportion to an optimal myopic action select. This is calculated by looking at ground truth and selecting the action that produces the best estimation at the next time step.

A GP regression model minimizing variance to select actions generated the most prediction error, an average 1.76x larger than optimal action selection using our CNN for modeling. The CNN based image inpainting network with random actions performed better, generating an error 1.17x worse then optimal. This demonstrates that regardless of how the samples are taken, the proposed CNN method can outperform a GP. Adding the proposed action selection improved the MSE, achieving a score of 1.13x optimal. A CNN trained and tested using Monte Carlo dropout as a representation of neural network uncertainty [20] was also evaluated, with actions selected to sample the most uncertain areas. This method's results are comparable to executing random actions.

TABLE II: The ratio of reconstruction error given a sampling strategy vs. sampling the most informative location reconstruction error on the MNIST data set. The robot collects 10 samples per action and is given a budget of 50 actions. Error bars show one SEM.

	GP	CNN-Random	CNN-Biased
MSE	$1.721 \pm .026$	$1.173 \pm .035$	$1.134 \pm .034$

B. Ocean Information Gathering

To evaluate our method on data from a representative robotic information gathering environment, we simulate an underwater vehicle as it gathers information in an ocean environment (e.g., for scientific data collection). The Regional Ocean Modeling System (ROMS) utilizes sophisticated models of ocean processes and forward simulations of prior data to generate a realistic estimate of ocean properties such as temperature, salinity and currents. The ROMS data we use consists of a 128x128 grid of ocean surface temperatures collected off Monterey Bay at a resolution of 300 m per data point. Data from May 1, 2016 to Feb 22, 2018, at 3:00, 9:00, 15:00 and 21:00 hours was aggregated and partitioned into a training set consisting of 1800 examples. Validation and test sets consist of 100 examples.

The same tests performed on the MNIST data are performed. The results can be viewed in Table III.

Table IV shows the results when actively choosing actions is simulated. The GP model generated the most prediction error, followed by our CNN approach of taking random action and minimizing the dropout estimated variance. Our proposed action selection technique generated the least prediction error.

TABLE III: The average temperature error between estimated data field and ground truth over 100 ocean data fields. The robot collects 10 samples per action and actions are taken at random. Budgets of 25, 50, 75, and 100 are shown. Error bars show one SEM.

	25	50	75	100 Actions
GP	$.634 \pm .031$	$.596 \pm .029$	$.564 \pm .028$	$.522 \pm .025$
CNN	$.533 \pm .026$	$.405 \pm .015$	$.347 \pm .018$	$.326 \pm .019$

TABLE IV: The ratio of reconstruction error given a sampling strategy vs. sampling the most informative location reconstruction error on the ocean data set. The robot collects 10 samples per action and is given a budget of 50 actions. Error bars show one SEM.

	GP	CNN-Random	CNN-Dropout	CNN-Biased
MSE	$1.94 \pm .09$	$1.32 \pm .05$	$1.29 \pm .05$	$1.20 \pm .04$

VI. Experiments, Results and Discussion - 3D

To show our proposed PCNN-LSTM layer and recurrent U-net architecture can generate quality time series estimates and produce informative action choices, we compare our method to Gaussian process regression in two different domains. The first domain is moving MNIST [21], a collection of videos of random MNIST digits bouncing around a frame. The second is time series ROMS data, a time series version of the data in the 2D case.

In addition to comparing the proposed method to the GP described above, we also compare to a modified 2D CNN approach. We use the same network described in Section IV-A.1, with an additional LSTM layer at the end, allowing it to learn from the temporal information. Comparing to this network should provide some insight into the importance of learning the spatial and temporal relationships of the data together, as opposed to learning spacial and temporal relationships independently. This network is referred to as the PCNN network.

A. Moving MNIST

We simulate a robot as it explores an environment represented by a moving MNIST dataset of 64x64 pixel. We start a simulated robot with spacially localized sensing in a random location at $t = 0$ and have it sample data in a 20x20 lawn mower pattern. This is because each sample will be taken at a unique time t, not allowing us to append samples taken at different times like was done in the 2D case. The samples taken at $t = 0$ are used by the neural network to estimate the data field for $t = 1$. The goal is to minimize the error between the estimated data field at $t + 1$ and the ground truth at $t + 1$.

The moving MNIST dataset requires quality sampling to be effective. Because the digit makes up a small percentage of the total video frame, and contains a majority of the information about the environment, sampling on the digit is extremely important to generating quality estimates. To evaluate reconstruction quality independent of sampling location, we sample the digit using the same sample location for the GP method, PCNN network and the proposed PCNN-LSTM network. The results can be seen in Table V. The results show that our proposed PCNN-LSTM network produces a per grid square error that is 30% lower than that produced by the GP method and 17% less than the PCNN method.

TABLE V: The average pixel error between estimated data field and ground truth over 100 moving MNIST videos of 30 frames. The robot collects 96 samples per action in a lawn mower pattern and actions are taken to maximize coverage of MNIST digit. Error bars show one SEM.

	GP	PCNN	PCNN-LSTM
Pixel Error	11.61 ± 2.86	9.75 ± 1.54	$8.12 \pm .82$

B. Time Series Ocean Data

We next test the proposed PCNN-LSTM reconstruction network on ROMS ocean data, predicting the ocean temperatures of the next frame given the 10 previous frames. The experimental setup is the same as with the moving MNIST dataset.

Table VI provides the results of 100 simulated trials of a robot compared to a GP. We measure a reduction in error of 52% compared to GP reconstruction and 25% compared to PCNN.

TABLE VI: The average temperature error between estimated data field and ground truth over 100 time series ocean videos of 30 frames. The robot collects 96 samples per action in a lawn mower pattern and actions are taken at random. Error bars show one SEM.

	GP	PCNN	PCNN-LSTM
Error (degrees)	$1.06 \pm .02$	$.67 \pm .02$	$.51 \pm .02$

VII. Conclusions

The results presented in this work demonstrate the power of using masked convolution neural networks to model and inform environments for robotic information gathering. This work shows that a CNN trained on historical data with biased loss functions can provide estimates that outperform with state-of-the-art GP techniques. Furthermore, we introduce a PCNN-LSTM layer and accompanying architecture for predicting time series environments where spatio-temporal information is linked. The method presented is generic and can be applied to any spatially correlated field for modeling or exploration.

References

[1] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, no. Feb, pp. 235–284, 2008.

[2] P. Whaite and F. P. Ferrie, "Autonomous exploration: Driven by uncertainty," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 193–205, 1997.

[3] C. Guestrin, A. Krause, and A. P. Singh, "Near-optimal sensor placements in gaussian processes," in *The 22nd international conference on Machine learning*. ACM, 2005, pp. 265–272.

[4] S. Bai, J. Wang, F. Chen, and B. Englot, "Information-theoretic exploration with bayesian optimization," in *Proceedings of International Conference on Intelligent Robots and Systems*. IEEE, 2016, pp. 1816–1822.

[5] R. Marchant and F. Ramos, "Bayesian optimisation for intelligent environmental monitoring," in *Proceedings of International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 2242–2249.

[6] D. Jayaraman and K. Grauman, "Learning to look around: Intelligently exploring unseen environments for unknown tasks," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE, 2018, pp. 1238–1247.

[7] A. F. Shchepetkin and J. C. McWilliams, "The regional oceanic modeling system (roms): a split-explicit, free-surface, topography-following-coordinate oceanic model," *Ocean modelling*, vol. 9, no. 4, pp. 347–404, 2005.

[8] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 417–424.

[9] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, pp. 341–346.

[10] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra, "Texture optimization for example-based synthesis," in *Transactions on Graphics (ToG)*, vol. 24, no. 3. ACM, 2005, pp. 795–802.

[11] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2536–2544.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[13] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *Transactions on Graphics (ToG)*, vol. 36, no. 4, p. 107, 2017.

[14] Y. Li, S. Liu, J. Yang, and M.-H. Yang, "Generative face completion," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 3911–3919.

[15] R. Yeh, C. Chen, T. Y. Lim, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with perceptual and contextual losses," *arXiv preprint arXiv:1607.07539*, vol. 2, p. 3, 2016.

[16] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[17] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," *arXiv preprint arXiv:1804.07723*, 2018.

[18] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[20] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016, pp. 1050–1059.

[21] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using LSTMs," in *Proceedings of International conference on machine learning*, 2015, pp. 843–852.