Self-Supervised Segmentation of River Scenes

Supreeth Achar, Bharath Sankaran, Stephen Nuske, Sebastian Scherer and Sanjiv Singh.

***Abstract*— Here we consider the problem of automatically and accurately segmenting visual images taken from a boat or low-flying aircraft. Such a capability is important for both mapping rivers as well as following rivers autonomously. The need for accurate segmentation in a wide variety of riverine environments challenges the state of the art in vision-based methods that have been used in more structured environments such as roads and highways. Apart from the lack of structure such as clearly defined road edges, the principal difficultly has to do with large spatial and temporal variations in the appearance of water in the presence of nearby vegetation and with reflections from the sky. We propose a self-supervised method to segment images into "sky", "river" and "shore" (vegetation + structures). Our approach uses simple assumptions of geometric structure common to rivers to automatically learn the relative importance of features such as color, texture and image location before they are used to segment the image. Our method performs robustly on three different image sequences from the Allegheny and Youghiogheny rivers with pixel classification error rates of less than 5%, outperforming a supervised method, trained and tested on the same data set. The performance of the proposed method seems sufficient to allow for guidance of an autonomous vehicle.**

I. Introduction

We are interested in a minimal sensor suite based on passive vision and inertial sensing that could be used to autonomously explore rivers, mapping their width as it proceeds. Given sensor pose and camera calibration it is straightforward to turn segmented images into river maps. In some cases, the canopy can be so thick and high around a river so as to block GPS signals and the problem becomes one of simultaneous localization and mapping. In any case, a key subproblem to be solved is to automatically segment visual images such that the extents of the river can be found accurately. Since rivers can look like roads, one idea is to use fairly well developed methods that have been used to track roads and highways with passive vision. Apart from the lack of structure such as clearly defined road edges, the principal difficultly of vision-based tracking of rivers has to do with large spatial and temporal variations in appearance of water in the presence of nearby vegetation and with reflections from the sky. See for example, Fig. 1 and Fig. 2.

Such problems are often solved with supervised learning, an approach where a system is trained based on representative data ahead of time. In our case, supervised learning would require that we train our river detector with instances of visual patches that correspond to water in rivers and then use the trained system online. Given the lack of a representation that would render the appearance of rivers as a constant even in varying lighting conditions, we find that it is necessary that any learning be self-supervised. That is, the system must train itself over time, adapting to local appearance as influenced by terrain, flora and lighting conditions. A natural question arises in being able to develop such a self-supervised method– whence the representation? There is a common intuition that features of color, texture and vertical location in an image might help. Still, at the outset it is not clear how to encode/combine different representations.

Here we propose a two step self-supervised method to perform such segmentation. In the first step we use a simple heuristic (knowledge of the horizon line) to automatically determine the relative correlation of between features (a grab bag of color and texture based features). This allows us to

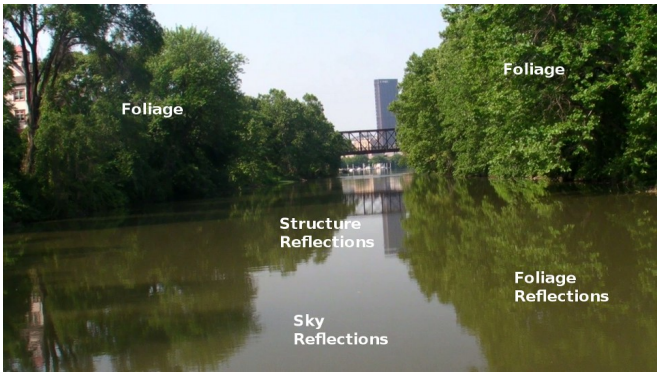


Fig. 1. An example image illustrating the variation of river appearance within a single image.

Fig. 2. This image was taken from almost the same location as Fig. 1 at a different time of day. The appearance of the river has changed dramatically.

S. Achar, S. Nuske, S. Scherer and S. Singh are with The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15232, U.S.A., {supreeth,nuske,basti,ssingh}@cmu.edu

B. Sankaran is with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA, 19104, U.S.A, bharath@seas.upenn.edu

score each image patch on its likelihood of belonging to a river. Patches with high certainties are used to train a Support Vector Machine. In the second step, the output of the SVM segments the image into distinct regions. The advantage of such an approach is that it assumes no prior to determine segmentation other than simple geometric assumptions such as that the river lies below the horizon, and, that river features are significantly different than other features below the horizon. All other training of features happens automatically as often as at every frame.

We have tested our algorithms with image sequences from two different rivers – the Allegheny and the Youghiogheny rivers in Pennsylvania, U.S.A.. The image sequences have distinctly different environmental conditions; sunny and dusk. We show how supervised learning (training on some of the sequences and testing on others) by presenting hand labeled instances of segmentation, provides a reasonable answer. We show how the results can be improved with our self-supervised method to the point that in our data sets it was common to get a misclassification rate around 2% and never higher than 5%.

In the following sections we review related work (Section II), show how the features are selected (Section III) and then present our algorithm (Section IV). The results are presented in Section V.

II. Related Work

The problem of segmenting a river scene from images can be thought of as a visual classification problem, where we classify the image into the classes; river and non-river (shore and sky regions). Classification techniques are categorized into two paradigms: supervised and self-supervised. Supervised problem setups rely on a labeled training set to make a global prediction on arbitrary images, while self-supervised approaches typically operate within one image to extrapolate information based on a self-generated training set from another source.

Some examples of supervised appearance classification and regression to infer scene structure are in Saxena et al. [2] and Hoiem et al. [3]. These approaches do not seem applicable to our work, because the appearance (color, illumination and texture) of a river is highly non-uniform both spatially across a single image and temporally from day to day and therefore it is most likely too challenging to learn a model offline which can encompass the large variety of appearance that rivers exhibit. Furthermore they are designed to work in general scenes, whereas we can exploit the general structure of a river scene to get higher accuracies.

Self-supervised classification, such as the road segmentation work of Dahlkamp et al. [1] collect training examples online from a portion of the current image (Dahlkamp et al. use the bottom of the image which is directly in front of the vehicle) and use to classify the remainder of the image. As is clear in Fig. 1 we need a different self-supervised approach for a river, because the areas of the river at the bottom of the image are vastly different from other areas of the river in the image.

Some researchers have specifically addressed the problem of visual water detection using a variety of different passive vision sensors; non-visible wavelength imagers, polarized imagery and hyperspectral imaging [4], [5], [6]. For reasons of cost and weight restrictions we consider a visible light camera solution.

The work in [7], [8], [9] uses conventional visible passive imagery to detect bodies of water. In [7] and [8] the authors detect water by fusing multiple cues like color, brightness and stereo depth to train a classifier. The recent work in [9] is for monocular vision and looks to generate a physical model of bodies of water lying on or nearby roads. They make the observation that in their application environment that the color of a water body tends to gradually change from the leading edge to the trailing edge (a property of incident angle), when other terrain types do not. They use this observation to detect water bodies in wide-open areas. Our application environment is quite different, a river scene that is not wide-open, variation in color of the river is far more dependent on the reflections of the surrounding environment as opposed to incident angle.

Rather than learn or derive a model offline of how the river will appear, we take an online approach that assumes a ground plane (river plane) and exploits the knowledge of the horizon to create self-supervised training sets of features within the current image.

III. Feature Selection

To enable effective river segmentation a discriminative feature vector ($X \in \mathbb{R}^d$) for describing local appearance is required. Color is a useful feature for detecting water as demonstrated in [7] & [8]. Texture is another useful cue, for example, parts of an image containing the river tend to be less textured than regions of foliage. Position in an image provides a strong prior, the higher up in an image a region is the less likely it is to be part of the river.

We selected features by empirically measuring the performance of different feature vectors describing color and texture. Each candidate feature vector was used to train a two class (river and shoreline) linear support vector machine over a small set of training images and then tested by using the SVM to segment out the river in images from a separate evaluation set. Performance of a feature vector was quantified by the percentage of correctly labeled pixels averaged over all the images in the evaluation set. All features were calculated over square 5×5 pixel patches.

For the color descriptor part of the feature vector we used the RGB, Lab and HSV color spaces individually and in various combinations to train classifiers. The performance of these classifiers on labeling river regions in the feature evaluation test set is show in Table I. A combination of RGB and Lab colorspaces gave the best performance and was selected as our color descriptor.

We evaluated the performance of three different texture filters using a similar methodology. The first was the response to a set of Laws' masks [10] over 4 different scales of the 3 channels of the Lab image. We used 8 of the 9 3×3 Laws'

Masks leaving out the mask that performs low pass filtering, so the resulting texture descriptor had length $4 \times 3 \times 8 = 96$. The second texture descriptor considered was the DAISY descriptor [11] without normalization. The third alternative was a bank of Gabor filters with 4 scales and 6 orientations. There was no significant difference in terms of performance between the three filter sets (Table I) so we chose the Laws' masks because it was the fastest to compute.

TABLE I
PERFORMANCE OF DIFFERENT FEATURE VECTORS FOR SEGMENTATION INTO TWO CLASSES

	Feature	Dimensionality	Error Rate
Color	RGB	3	47.05%
	Lab	3	35.61%
	HSV	3	46.62%
	RGB+HSV	6	44.43%
	RGB+Lab	**6**	**32.90%**
	RGB+Lab+HSV	9	37.60%
Texture	**Laws' Masks**	**96**	**26.81%**
	DAISY	200	26.76%
	Gabor	24	27.19%
Combined	Color+Texture	102	19.73%
	Color+Texture+Height	**103**	**4.61%**

When combined, color and texture features perform better than either cue alone. Image position, specifically the height of a region in image coordinates, provides a strong contextual cue for segmentation and hence is included in the feature vector. The bottom of Table I shows the labeling error rate for the combined color and texture descriptor and for our final choice of feature descriptor on the feature evaluation set.

IV. SELF-SUPERVISED RIVER SEGMENTATION

Riverine environments vary widely in appearance which makes building a single classifier that works reliably in different conditions difficult. Such a classifier would require labeled training images captured in a variety of environments and under different conditions. Its performance in an environment dissimilar to those seen earlier would not be assured. Instead of attempting to learn offline a universal model of river appearance, we automatically learn a new appearance model online for each image presented to the algorithm.

To make this possible we utilize knowledge about the position of the horizon in each image and assume that anything appearing above the horizon line is not part of the river. In our application domain, the inertial measurement unit and altimeter on the air vehicle would be used to calculate the horizon line. In the absence of sensor data, a vision based estimate of the horizon can be used instead [12]. The assumption that the river can not be above the horizon is valid except for situations in which the river has a significant upward slope (these are unusual scenarios in which we do not expect to operate like upstream on a rapid or at the base of a waterfall). An additional assumption we make is that the appearance of the above horizon regions is fairly representative of the appearance of ther entire shore area in that image.

As a preprocessing step, we first segment out the sky and ignore it during all further computations. To build an online appearance model for discriminating between river and shore regions, we automatically extract two types of patches from the image, those that are part of the shore region and those that are likely to be part of the river. Generating shore patches is trivial, we just sample patches lying above the horizon line. We find the patches below the horizon line that are most dissimilar in appearance to those above the horizon and use them as candidate river patches. We then use these patches as training examples for a linear SVM classifier which is then used to classify all parts of the image. Details of each step in the algorithm are given below.

A. Feature Extraction

As described in the Section III, the image is divided into square patches and a feature vector ($X \in \mathbb{R}^n$) is computed for each patch which describes its color, texture and position.

B. Sky Detection

A linear support vector machine trained over a set of labeled images is used to detect the sky region in each image. The sky is relatively easy to segment out with a globally trained classifier as its appearance is not as variable as that of the river. Discarding the sky before further processing reduces computational effort and prevents confusion with shiny, mirror like parts of the water's surface which often have an appearance similar to that of the sky. Fig. 3(a) shows an example of sky detection. The remainder of the image needs to be classified as either being part of the river or part of the shore.

C. Appearance Modeling

The goal of the appearance modeler is to assign to each patch a probability of being part of the river (R) or being part of the shore ($\neg R$) on the basis of the feature vector (X) that describes it. By Bayes' Rule we have

$$P(\neg R \mid X) = \frac{P(X \mid \neg R)P(\neg R)}{P(X)} \tag{1}$$

Since the labels (R and $\neg R$) are what we are trying to determine, $P(X \mid \neg R)$ can not be calculated directly. We define a new boolean variable H which is true for regions below the horizon line and false for regions above the horizon. The value of H for each position in the image is known because the horizon is known. We assume that the appearance of the shore regions above the horizon is representative of the entire shore region in the image, this implies that $P(X \mid \neg R) \approx P(X \mid \neg H)$ which gives

$$P(\neg R \mid X) \approx \frac{P(X \mid \neg H)P(\neg R)}{P(X)} \tag{2}$$

$$= \frac{P(X \mid \neg H)P(\neg R)}{P(X \mid \neg H)P(\neg H) + P(X \mid H)P(H)} \tag{3}$$

The appearance distributions $P(X \mid \neg H)$ and $P(X \mid H)$ are modeled using Chow Liu trees [13]. A Chow Liu tree is a method for approximating the joint probability distribution of a set of discrete random variables by factorizing it into a product of second order distributions. A Chow Liu tree is optimal in the sense that it is the distribution of its class that minimizes the KL divergence to the real distribution.

The feature vector X contains color, texture and image position information and has high dimensionality as it contains many texture filter responses. For the Chow Liu appearance modeling an abridged feature vector $\hat{X} \in \mathbb{R}^d$ is created in which the texture descriptor subvector is replaced by its L_2 norm because using the full length feature vector would slow down computation. Additionally, $\hat{X}$ does not include image position. Since we are modeling only appearance including position information in $\hat{X}$ would introduce an undesirable bias.

The features in feature vector $\hat{X}$ are continuous valued while Chow Liu trees work over discrete valued random variables. To solve this problem, $\hat{X} \in \mathbb{R}^d$ for each image patch is converted into a discretized feature vector $\tilde{X} \in \mathbb{S}^d$ where each feature in $\tilde{X}$ is assigned to 1 of 16 levels ($\mathbb{S} = \{0,1,2,\ldots,15\}$) using equally sized bins that span the range of values taken by that feature.

Two Chow Liu trees are built, one using the the feature vectors of patches above the horizon ($\tilde{X} \notin H$) to model $P(X \mid \neg H)$ and another using the feature vectors of below horizon patches ($\tilde{X} \in H$) to model $P(X \mid H)$. We use subscripts to denote individual features in a feature vector so $\tilde{X}_i$ is the i^{th} feature in feature vector $\tilde{X}$. A Chow Liu tree is a maximal spanning tree of the mutual information graph. The mutual information graph has one node for each feature ($\tilde{X}_i$) and an edge between every pair of nodes ($\tilde{X}_i$ and $\tilde{X}_j$) which is weighted by their mutual information $I(\tilde{X}_i; \tilde{X}_j)$.

$$I(\tilde{X}_i; \tilde{X}_j) = \sum_{x_i=0}^{|\mathbb{S}|-1} \sum_{x_j=0}^{|\mathbb{S}|-1} p(x_i,x_j) \log\left(\frac{p(x_i,x_j)}{p(x_i)p(x_j)}\right) \quad (4)$$

Where $p(x_i)$ is the probability that the i^{th} feature in $\tilde{X}$ takes on the value x_i, $P(\tilde{X}_i = x_i)$. Similarly, $p(x_j)$ is $P(\tilde{X}_j = x_j)$ and $p(x_i,x_j)$ is the joint probability distribution $P(\tilde{X}_i = x_i, \tilde{X}_j = x_j)$. All these distributions are estimated directly from $\tilde{X} \notin H$ and $\tilde{X} \in H$ by counting feature value occurrences and co-occurrences. Edges are deleted from the mutual information graph to form a maximal spanning tree. The resulting tree encodes the approximate distribution as a product of pairwise conditionals. We can calculate $P(\tilde{X} = \tilde{x})$, the probability of occurrence of a particular feature vector $\tilde{x} \in \mathbb{S}^d = \{\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \ldots, \tilde{x}_d\}$ as follows

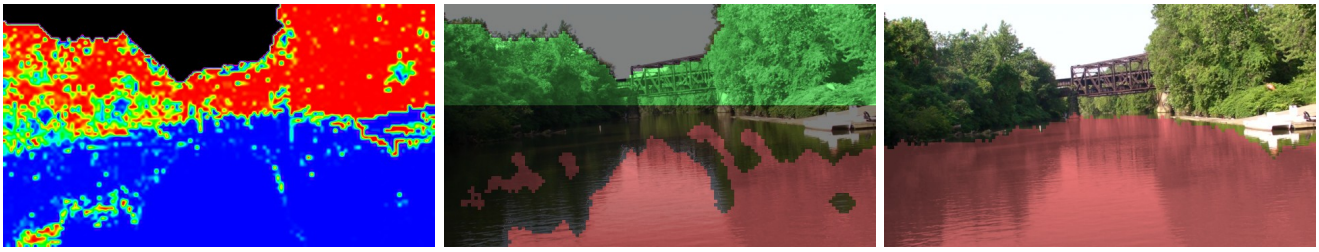
$$P(\tilde{X} = \tilde{x}) \approx \prod p(\tilde{X}_i = \tilde{x}_i \mid \tilde{X}_{\pi(i)} = \tilde{x}_{\pi(i)}) \quad (5)$$

where $\pi(i)$ is the parent of node $\tilde{X}_i$ in the tree. The two Chow Liu trees for $P(X \mid \neg H)$ and $P(X \mid H)$ are used in (2) to calculate $P(\neg R \mid X)$ for each patch in an image. An example is shown in Fig. 3(a).

D. Classifier Training

For each patch, the probability of being part of the shore region $P(\neg R \mid X)$ is calculated using (2). The patches that are least likely to be on the shore ($P(\neg R \mid X) < T$) are used as candidate river patches. Using a low value for T reduces the chance that shore patches are accidentally used as candidate river patches, but if T is set too low then the selected region will be too small to provide a good representation of the river region. In our experiments, T was set to 0.01. The shore patches above the horizon and the candidate river patches with $P(\neg R \mid X) < T$ are used to train a two class (river/shore) linear support vector machine. Fig. 3(b) shows the selected river and shore training examples for an input image.

The SVM uses the unabridged, continuous valued feature vectors X, including image position and the complete texture descriptor. The support vector machine is trained using an online stochastic gradient descent method. Since we are learning a new classifier for each new frame, while the appearance of the river is likely to remain fairly consistent over short periods of time, we initialize the SVM training using the SVM model learnt on the previous frame. This reduces the number of gradient descent iterations needed to train the classifier for each new frame. Fig. 3(b) shows the river and shore training examples picked by the algorithm in an image.



(a) $P(\neg R \mid X)$ (b) Training Examples (c) Final Result

Fig. 3. Steps in the Self-Supervised River Segmentation Algorithm: **(a)** $P(\neg R \mid X)$: warm colors are used for values closer to 1 (probably shore region). The detected sky region is marked in black **(b)** Training examples: The algorithm finds shore regions (marked in green) and candidate river regions (marked in red) for training a classifier. **(c)** Final result: The classifier is run on the entire image to classify each patch. The detected extent of the river is marked in red.

E. Detecting Water

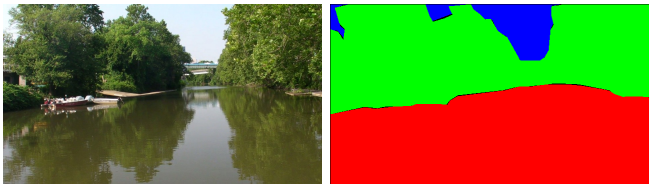
The learnt SVM model is used to classify all the patches in the image. As a post processing step, small holes in the labeling are filled using morphological operators. An example final segmentation result is shown in Fig 3(c)

V. Results

A. Datasets and Groundtruth

Three datasets were used for evaluating our algorithm. The data collection setup used was a tripod mounted camera (a SANYO VPC-CG102BK for the *Allegheny Day* dataset discussed following and a Canon SX200IS for the other two datasets) placed onboard a small motorboat. The cameras captured 1280×720 images which were downsampled to 640×360 for processing. Two of the datasets were collected on the Allegheny river at Washington's Landing in Pittsburgh, PA, U.S.A, the third was collected on the Youghiogheny at Perryopolis, PA, U.S.A.. One of the Allegheny datasets (referred to as Allegheny Day) was collected during the afternoon and the other (Allegheny Dusk) was collected in the evening close to dusk. The dataset on the Youghiogheny was collected around noon. Each dataset contained between 120 and 150 images that were manually labeled to provide groundtruth for performance evaluation. The labels used were river, shore (vegetation, structures etc.) and sky. An example ground truth labeling is shown in Fig. 4.

The performance metric used in all tests and while reporting results is the percentage of pixels misclassified by the algorithm when compared against ground truth. Since the classifiers used generate only two output labels and we are not interested in differentiating between the shore region and sky we treat them as a both as a single class (non-river) during performance evaluation.



(a) (b)

Fig. 4. **(a)** an image from the Allegheny Day dataset and **(b)** its groundtruth labeling with the river marked in red, shore (foliage/structures etc.) colored green and the sky in blue.

B. Supervised Segmentation Results

We investigated how well a supervised classifier would be able to generalize to images from previously unseen environments. For each dataset two classifiers were built, the first classifier (Self in Table II) was trained using two randomly selected images from the same dataset and the second classifier (LOO: Leave one out) was trained on two images selected from the other two datasets. The error rates for these classifiers were averaged over 16 trials and are tabulated in Table II. It can be seen that the supervised approach works well when used on datasets it had been trained on (the self column) but performance degrades when run on the other two datasets that were not seen during training (Leave one out). This suggests that a supervised approach does not generalize well to new environments.

TABLE II
Supervised Segmentation Performance: Mean Error(Std dev)

	Self	LOO
Allegheny Day	3.77% (0.64%)	4.01% (0.63%)
Allegheny Dusk	3.55% (0.46%)	8.13% (1.53%)
Youghiogheny	3.50% (0.34%)	4.47% (0.41%)

C. Self-supervised Segmentation Results

We evaluated the self-supervised segmentation algorithm described in Section IV on the three datasets. Error rates are shown in Table III. On all three datasets, the self supervised algorithm outperformed a supervised classifier that was tested on previously unseen datasets. Even when the supervised algorithm was trained on a subset of images from the same dataset it was tested on, it only outperformed the self supervised method on Allegheny Dusk. This is significant considering how the self-supervised method uses no manually labeled input.

Fig. 5 shows some example images from the three datasets and the detected extent of the river along with the major intermediate outputs of the algorithm. Some images on which our algorithm failed to detect the river correctly are shown in Fig. 6. In Fig. 6(a) part of the tree was misclassified because it was in the shadows and also parts of the background just above the horizon were incorrectly labeled as parts of the river. In Fig. 6(b) a large swath of the grass on the bank was marked as part of the river because a few grass patches were selected as positive river examples for the classifier.

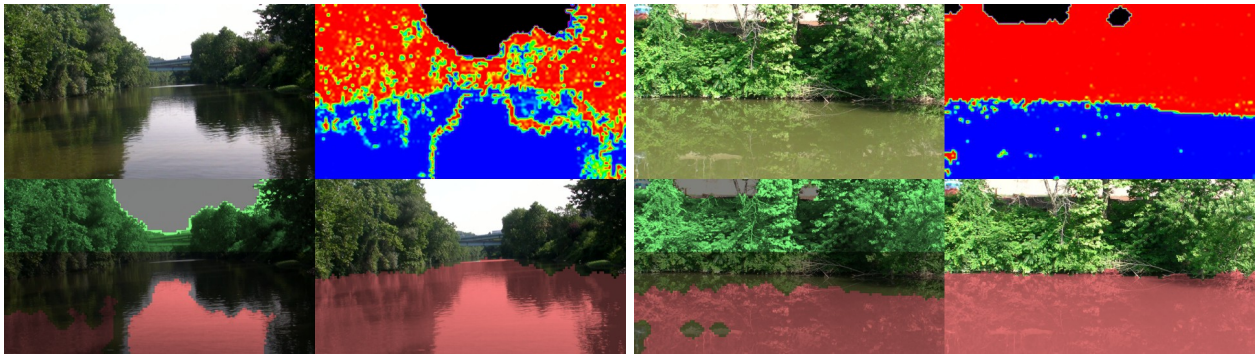
TABLE III
Self Supervised Segmentation Performance

Location	Mean Error Rate
Allegheny Day	2.71%
Allegheny Dusk	4.15%
Youghiogheny	2.67%

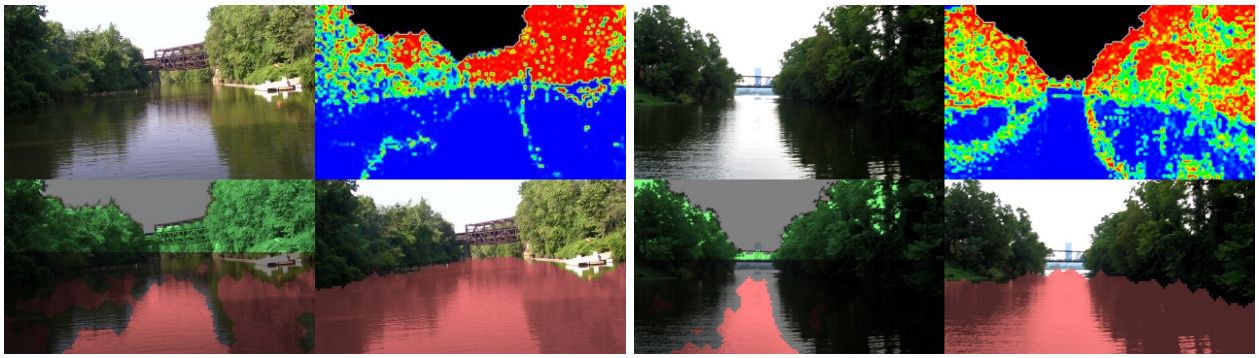
VI. Conclusions and Future Work

We presented a method for using monocular vision to estimate the extent of a river in an image. We demonstrated that a naive supervised technique is unlikely to generalize well to different environments. We formulated and evaluated a self-supervised alternative that automatically generates a model of river appearance when presented with an image by leveraging assumptions that can be made about the structure of the environment and the horizon.

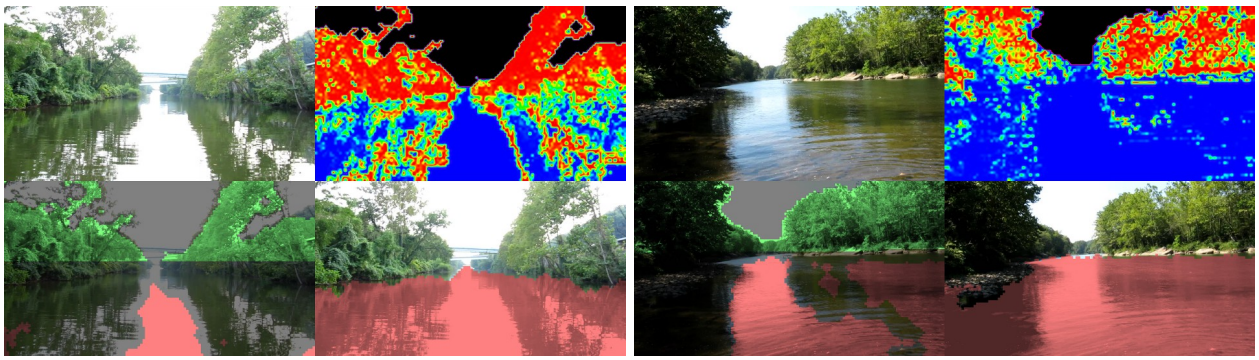
Our current system is completely memoryless as it builds a new river appearance model for every input image. Using some history of previous river appearance to build an



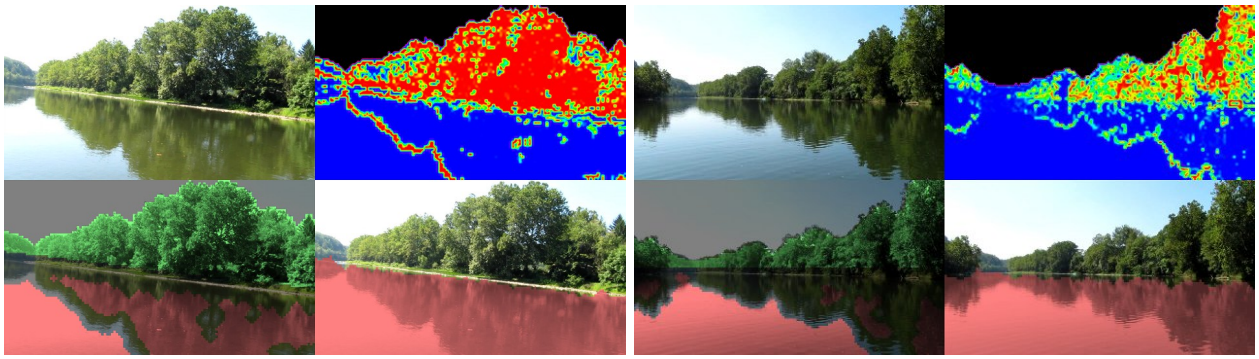
(a) Allegheny Day (b) Allegheny Day



(c) Allegheny Day (d) Allegheny Dusk



(e) Allegheny Dusk (f) Youghiogheny



(g) Youghiogheny (h) Youghiogheny

Fig. 5. Example frames from different datasets on which the self supervised algorithm performed well. For each example the top left quadrant is the input, the top right quadrant is a heatmap showing the probability of a patch being part of the shore based on its appearance (the regions detected as part of the sky are marked black), the bottom left quadrant shows the automatically detected river and shore regions for training a classifier and the bottom right quadrant shows the extent of the river as determined by the algorithm.

adaptive model could increase performance and provide robustness against failure. The most serious failure mode in our algorithm is when novel objects (boats, piers etc) appear below the horizon but do not get marked as non-river. Our algorithm is unable to handle this adequately for two reasons. Firstly it does not maintain a model of river appearance and secondly it assumes that the appearance of the above horizon region adequately models all non-river objects. Keeping a model of river appearance learnt from previously seen images would allow the algorithm to detect novel non-river objects below the horizon.

We would also like to investigate the use of priors on likely river shapes and performing inference over multiple frames in a video sequence to increase accuracy.

REFERENCES

[1] H. Dahlkamp, A. Kaehler, D. Stavens, S. . Thrun, and G. . Bradski, "Self-supervised monocular road detection in desert terrain," in *Proc. Robotics Science and Systems Conference (RSS'06)*, Philadelphia, PA, Aug. 2006.

[2] A. Saxena, M. Sun, and A. Ng, "Learning 3-d scene structure from a single still image," *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8, 2007.

[3] D. Hoiem, A. A. Efros, and M. Hebert, "Automatic photo pop-up," *Proceedings of ACM SIGGRAPH 2005*, Jan 2005.

[4] A. Sarwal, J. Nett, and D. Simon, "Detection of small water bodies," in *PrecepTek Robotics Technical Report (ADA433004'04)*, Littleton, CO, 2004.

[5] B. Xie, Z. Xiang, H.Pan, and J.Liu, "Polarization-based water hazard detection for autonomous off-road navigation," in *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS'07)*, San Diego, CA, Sept. 2007.

[6] H. Kwon, D. Rosario, N. Gupta, M. Thielke, D. Smith, P. Rauss, P. Gillespie, and N. Nasrabadi, "Hyperspectral imaging and obstacle detection for robotics navigation," U.S Army Research Laboratory Technical Report, Tech. Rep. (ARL-TR-3639'05), Sept. 2005.

[7] A. Rankin, L. Matthies, and A. Huertas, "Daytime water detection by fusing multiple cues for autonomous off-road navigation," in *Proc. 24th Army Science Conference (ASC'04)*, Orlando, FL, Nov. 2004.

[8] A. Rankin and L. Matthies, "Daytime water detection and localization for unmanned ground vehicle autonomous navigation," in *Proc. 25th Army Science Conference (ASC'06)*, Orlando, FL, Nov. 2006.

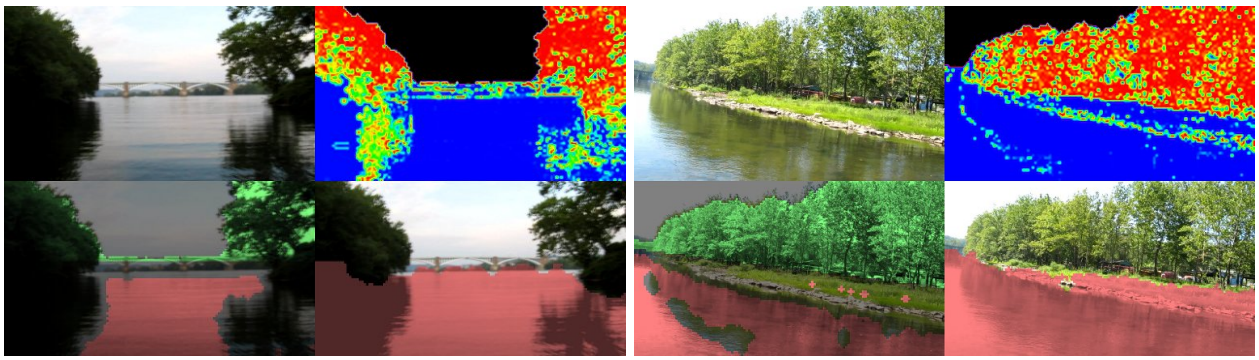
[9] ——, "Daytime water detection based on color variation," in *Proc.IEEE International Conference on Intelligent Robots and Systems (IROS'10)*, Taipei, Taiwan, Oct. 2010.

[10] K. Laws, "Rapid texture identification," *SPIE*, pp. 376–380, 1980.

[11] E. Tola, V. Lepetit, and P. Fua, "A fast local descriptor for dense matching," in *Conference on Computer Vision and Pattern Recognition*, Alaska, USA, 2008.

[12] S. Ettinger, M. Nechyba, P. Ifju, and M. Waszak, "Towards flight autonomy: Vision-based horizon detection for micro air vehicles," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Jan 2002.

[13] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," *Information Theory, IEEE Transactions on*, vol. 14, no. 3, pp. 462 – 467, may. 1968.



(a) Allegheny Dusk (b) Youghiogheny

Fig. 6. Some images on which the self supervised algorithm failed. In **(a)** the base of the tree and some regions just above the horizon line are marked as part of the river. In **(b)** the grass on the bank is marked as part of the river because a few patches on the bank were picked as river training examples.