

Causal Video Object Segmentation From Persistence of Occlusions

SUPPLEMENTARY MATERIAL

1. Additional results

In the main paper, we reported results on the BVSD dataset. The dataset contains three “subtasks”—“motion”, “camera motion”, and “non-rigid motion” (details in [3]). Results on these subtasks are reported below in Fig. 1 and Table 1.

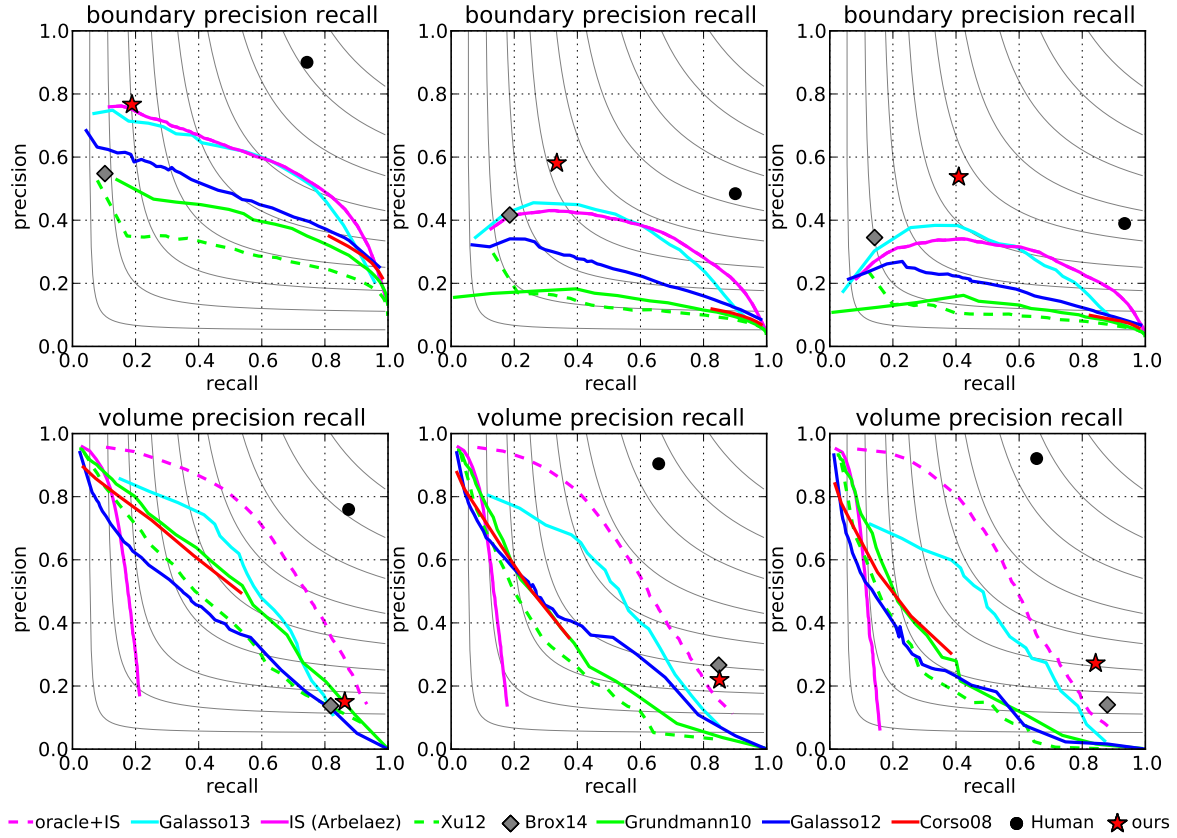


Figure 1: Results on BVSD subtasks; from left to right: “motion”, “camera motion”, “nonrigid motion”

	BPR			VPR		
	P	R	F	P	R	F
Motion	58.07	33.50	42.48	21.93	85.04	34.87
Camera motion	76.65	18.87	30.28	14.99	86.28	25.55
Nonrigid motion	53.71	40.86	46.42	27.20	84.45	41.12
General	76.02	18.65	29.95	13.56	87.09	23.46

Table 1: Precision-recall statistics of our method on BVSD subtasks. “General” subtasks contains all sequences and was reported in main paper.

2. Optimization details

In this section we derive the prox-operators used by the primal-dual algorithm.

2.1. Proximal operator for $G(c) = \tau^T c + \kappa^T \max(0, 1 - c) + I_{\{c \geq 0\}}(c)$

First define $g_i(c_i) = \tau_i c_i + \kappa_i \max(0, 1 - c_i) + I_{\{c_i \geq 0\}}(c_i)$, so that $G(c) = \sum_{i=1}^n g_i(c_i)$. This is done in anticipation of the prox-operator for G being separable¹:

$$\mathbf{prox}_{\sigma G}(y) = \arg \min_{c \geq 0} \frac{1}{2\sigma} \|c - y\|_2^2 + \tau^T c + \kappa^T \max(0, 1 - c) \quad (1)$$

$$= \arg \min_{c \geq 0} \sum_{i=1}^n \frac{1}{2\sigma} (c_i - y_i)^2 + \tau_i c_i + \kappa_i \max(0, 1 - c_i) \quad (2)$$

$$= (\mathbf{prox}_{\sigma g_1}(y_1), \dots, \mathbf{prox}_{\sigma g_n}(y_n)) \quad (3)$$

We now will derive the expression for $\mathbf{prox}_{\sigma g_i}(y_i)$. Since the general form of the expression will be the same for each i , we drop the index for clarity. For convenience, let $f(c) = \frac{1}{2\sigma} (c - y)^2 + g(c)$ (as shown in Fig. 2, f is a parabola with a kink), i.e. $\mathbf{prox}_{\sigma g}(y) = \arg \min f(c)$. Since f is strictly convex and the minimizer is unique, we initially ignore the constraint $\{c \geq 0\}$, eventually projecting the minimizer back onto the feasible domain if necessary. The subdifferential of $f(c)$ is

$$\partial f(c) = \begin{cases} \frac{1}{\sigma}(c - y) + \tau & \text{if } c > 1 \\ \frac{1}{\sigma}(c - y) + \tau - \kappa & \text{if } 0 \leq c < 1 \\ \frac{1}{\sigma}(c - y) + \tau - \kappa[0, 1] & \text{if } c = 1 \end{cases} \quad (4)$$

The optimality condition $0 \in \partial f(c)$ is satisfied when

$$c^* = \begin{cases} y - \sigma\tau & \text{if } y > 1 + \sigma\tau \\ 1 & \text{if } y \in [1 + \sigma(\tau - \kappa), 1 + \sigma\tau] \\ \max(0, y - \sigma(\tau - \kappa)) & \text{if } y < 1 + \sigma(\tau - \kappa) \end{cases} \quad (5)$$

These three cases are shown in Fig. 2,

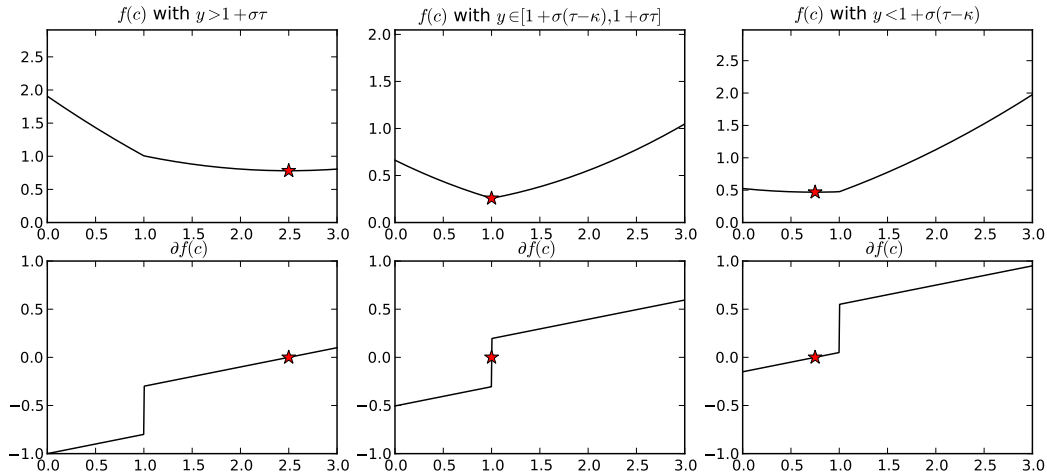


Figure 2: Top: $f(c)$ shown for the three cases described in (5). Bottom: subdifferentials of corresponding $f(c)$. Shown in red are the pairs $(c^*, f(c^*))$.

¹We use the following “separable sum” rule. If $f(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}) = g(x_1) + h(x_2)$, then $\mathbf{prox}_f(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}) = \begin{pmatrix} \mathbf{prox}_g(x_1) \\ \mathbf{prox}_h(x_2) \end{pmatrix}$

which can be rewritten as

$$\mathbf{prox}_{\sigma g_i}(y_i) = \max \left(0, \min \left(y_i - \sigma(\tau_i - \kappa_i), \max \left(1, y_i - \sigma\tau_i \right) \right) \right). \quad (6)$$

Using (3) and interpreting max/min componentwise, the prox-operator for G is written as

$$\mathbf{prox}_{\sigma G}(y) = \max \left(0, \min \left(y - \sigma(\tau - \kappa), \max \left(1, y - \sigma\tau \right) \right) \right). \quad (7)$$

2.2. Proximal operator for $F_1^*(y)$, with $F_1(y) = \|Wy\|_1$

We note that this operator appears in all problems that involve TV-regularization, that our derivation below is by no means novel, and that it is included only for completeness. Since the function is a (weighted) norm, we expect its convex conjugate (i.e. F_1^*) to be a (weighted) indicator of a dual norm ball. Moreover, we expect the proximal operator of the indicator function to be the projection onto the feasible set [2]. The steps below verify it.

$$F_1^*(z) = \sup_y y^T z - F_1(y) \quad (8)$$

$$= \sum_{i=1}^n \max_{y_i} \underbrace{y_i z_i - w_i |y_i|}_{g_i(y_i)} \quad (9)$$

The maximum of each term $g_i(y_i)$ can be determined as

$$\max_{y_i} g_i(y_i) = w_i \left(\max_{y_i} y_i \frac{z_i}{w_i} - |y_i| \right) \quad (10)$$

$$= \begin{cases} 0 & \text{if } \left| \frac{z_i}{w_i} \right| \leq 1 \\ +\infty & \text{else} \end{cases}, \quad (11)$$

so $F_1^*(z)$ is the indicator of the weighed ℓ_∞ ball:

$$F_1^*(z) = \begin{cases} 0 & \text{if } \|W^{-1}z\|_\infty \leq 1 \\ +\infty & \text{else} \end{cases}. \quad (12)$$

Using $\text{diag}(W)$ to write the main diagonal as a vector (i.e. $\text{diag}(W) = (W_{11}, \dots, W_{nn})$), the prox operator for F_1^* can be written as

$$\text{prox}_{\sigma F_1^*}(y) = \arg \min_z \frac{1}{2\sigma} \|z - y\|_2^2 + F_1^*(z) \quad (13)$$

$$= \text{sign}(y) \min\{\text{diag}(W), |y|\}. \quad (14)$$

2.3. Proximal operator for $F_2^*(y)$, with $F_2(y) = \lambda^T \max(0, 1 - y)$

The convex conjugate is

$$F_2^*(z) = \max_y z^T y - F_2(y) \quad (15)$$

$$= \sum_{i=1}^n \max_{y_i} \underbrace{y_i z_i - \lambda_i \max(0, 1 - y_i)}_{g_i(y_i)}. \quad (16)$$

Notice that when $z_i > 0$, $\max_{y_i} g_i(y_i) \rightarrow \infty$ (achieved with $y_i \rightarrow \infty$) and similarly $\max_{y_i} g_i(y_i) \rightarrow \infty$ when $z_i < -\lambda_i$ (achieved with $y_i \rightarrow -\infty$). These cases are shown in Fig. 3. So, $F_2^*(z) = \infty$ for $z \notin [-\lambda, 0]$. For $z \in [-\lambda, 0]$ we compute the subdifferential of $g_i(y_i)$ (both shown in Fig. 3):

$$\partial g_i(y) = \begin{cases} z_i + \lambda & y_i < 1 \\ z_i + \lambda[0, 1] & y_i = 1 \\ z_i & y_i > 1 \end{cases}. \quad (17)$$

The optimality condition $0 \in \partial g_i(y_i^*)$ suggests that when $z_i \in (-\lambda_i, 0)$, $y_i^* = 1$. In other words, for that interval, we have $\max_{y_i} g_i(y_i) = z_i$. On the interval boundaries y^* is not unique (when $z_i = -\lambda$, $y^* \in (-\infty, 1]$, and when $z_i = 0$, $y^* \in [1, \infty)$), but $\max_{y_i} g_i(y_i) = z_i$ still holds. So the convex conjugate can be written as follows:

$$F_2^*(z) = \begin{cases} 1^T z & \text{if } z \in [-\lambda, 0] \\ +\infty & \text{else} \end{cases}. \quad (18)$$

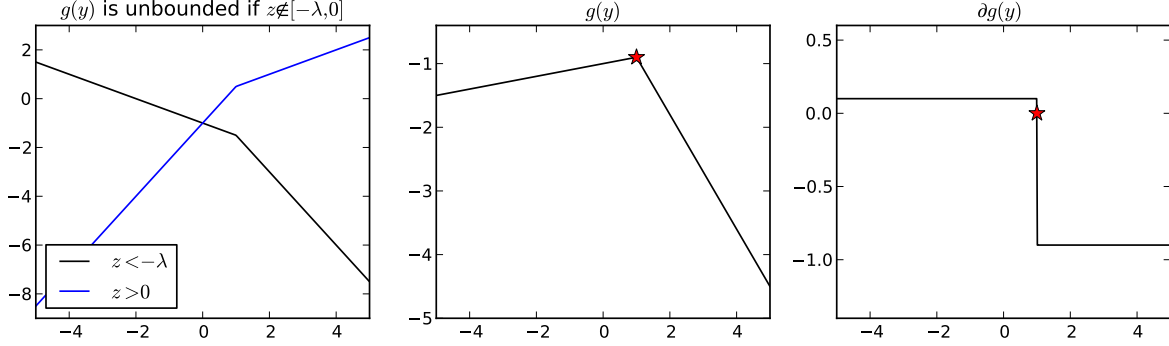


Figure 3: Left: $g(y)$ plotted for $z \notin [-\lambda, 0]$. As noted in text, these functions are unbounded. Middle: $g(y)$ with $z \in (-\lambda, 0)$. This is a piecewise linear function with a unique maximizer (if $z \in \{-\lambda, 0\}$, the function is bounded, but the maximizer is not unique). Right: subdifferential of $g(y)$ on the left.

To solve for $\text{prox}_{\sigma F_2^*}(z)$, we first find $\arg\min_x \frac{1}{2\sigma} \|z - x\|_2^2 + 1^T x$ (ignoring the constraint $x \in [-\lambda, 0]$), and then project it onto the feasible set:

$$\text{prox}_{\sigma F_2^*}(z) = \arg \min_x \frac{1}{2\sigma} \|z - x\|_2^2 + F_2^*(x) \quad (19)$$

$$= \min \left(\max(z - \sigma 1, -\lambda), 0 \right). \quad (20)$$

2.4. Layer unity prior and aggregated TV weights

In section 3.2 we made a claim that enforcing $\nabla c(x) = 0$ with a hinge loss penalty and associated cost $u(x)$ (nonzero for x where we want to enforce the constraint, and is zero elsewhere) is equivalent to increasing weights in TV regularization. We prove this claim here.

$\nabla c(x) = 0 \Leftrightarrow |\nabla c(x)| \leq 0$. Relaxed as a hinge loss penalty, this becomes

$$\int_D u(x) \max(0, |\nabla c(x)|) dx = \int_D u(x) |\nabla c(x)| dx. \quad (21)$$

But this is just TV regularization, and the objective already includes a penalty of the same form with weights $g(x)$. We conclude that the new “biased” penalty is $\int_D g'(x) |\nabla c(x)| dx$ with $g'(x) = g(x) + u(x)$.

3. Flow extrapolation via cross bilateral filter

In this section, we show additional examples of the result of flow extrapolation in the occluded region, which allows us to associate occluder points to occluded points. In each figure, the estimated motion field (v_t^{t+1}) and the extrapolated motion field (\hat{v}_t^{t+1}) are shown, below which regions of notable change are shown in individual in panels for closer inspection. In most cases, the extrapolation step locally improves optical flow.

For the car in Fig. 4, panels A-F show signs of improvement, where the flow in the occluded region becomes more similar to the “locally background” object. We note specific improvements as follows: **A**: the rear-view mirror is obtained. **B**: the motion in the region to the left of the car windshield becomes more similar to the background motion. **C**: incorrect optical flow in front of the car is fixed. **D-E**: most of the occluded region’s motion (in front of and below the car) becomes more similar to the background’s. **F**: the flow of the car wheel is independent of the vehicle motion, but is smoothed out. For the task of object segmentation at this scale, this is desired behaviour.

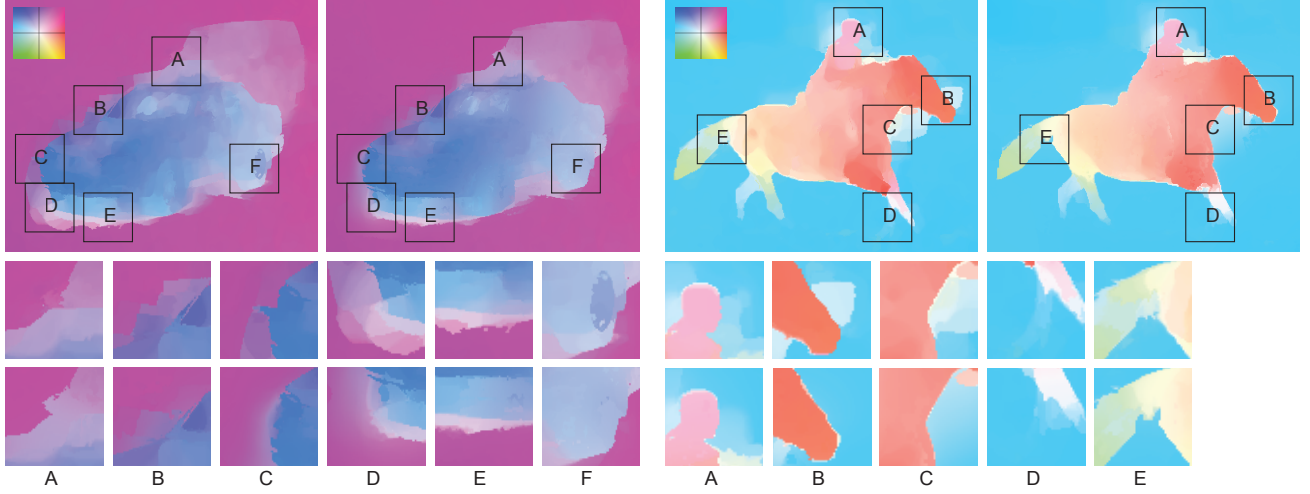


Figure 4: Two examples comparing the original optical flow field (v_t^{t+1}) (top left) and the resulting flow post-extrapolation (\hat{v}_t^{t+1}) (top right). Below each pair are panels highlighting regions where significant change occurred. For each box, v_t^{t+1} is shown in the top row while the extrapolated result (\hat{v}_t^{t+1}) is shown below it.

For the horse in Fig. 4, panels A-C show improvement. Box D is a failure example: extrapolated flow is worse than the original. Box E shows negligible improvement. Details follow: A: optical flow in the region around the head becomes similar to background. B: the erroneous motion in front of the horse’s nostril is removed. C: motion in front of the horse’s neck becomes is cleaned up. D: the left front leg of the horses is smoothed out, as the motion and appearance of are similar to background. When this occurs, the flow smoothes across the weak flow boundaries. E: the details on the tail improve slightly.

3.1. Constraint perturbation

The ability to compute occluded-occluder pairs (y^c, y) reliably requires accurate optical flow, which may be difficult to obtain, especially in regions with large motion. Inaccurate flow may lead to an incorrect constraint (e.g. both points falling on the occluded region). To be robust against failure of optical flow, we assume that depth layers can be locally discriminated by their appearance, and locally perturb the constraints to ensure that both points fall on their respective sides of the occlusion boundary.

Specifically, we model the local appearance of local foreground (*occluder*) and local background (*occluded*) regions by a pair of Gaussian mixture models (GMM) with respective likelihoods f_F and f_B . Learning these models would be simple if we could obtain a set of samples from F and B regions. Since these are not known, we estimate f_F and f_B from nearby “occluder” and “occluded” points. We found that this approach works well in practice, despite the implicit assumption that the majority of these points fall into correct regions. To measure how well a point matches the appearance of *occluder* and *occluded* regions, we introduce likelihood ratios $\phi_F(x) \doteq \log \frac{f_F(I(x))}{f_B(I(x))}$ and $\phi_B(x) \doteq -\phi_F(x)$. If $\phi_F(x) > 0$ (resp. < 0), a point is likely in the foreground region (resp. in background region).

We would like to transform the pair (y^c, y) to ensure that y^c lies in foreground region and that y lies in background region, while penalizing large transformations. To achieve this, the procedure tractable, we restrict the transformation to be parameterized by translation, rotation, and uniform scale (i.e. a *similarity*). Formally, we can write down an optimization problem:

$$\max_g \phi_F(g \circ y^c) + \phi_B(g \circ y) - \|g\|, \text{ with } g \text{ a similarity transformation} \quad (22)$$

where we use the notation $g \circ x$ to denote a point x transformed by g . The first two terms measure how well the image intensities near transformed points match the appearance models. The deformation penalty is denoted by $\|g\|$. Once the best transformation g^* is found, we can use the transformed constraint pair ($g^* \circ y^c, g^* \circ y$). In practice, we compute the value of (22) for multiple local transformations, and choose the best one. An example of this procedure is shown in Fig. 5.

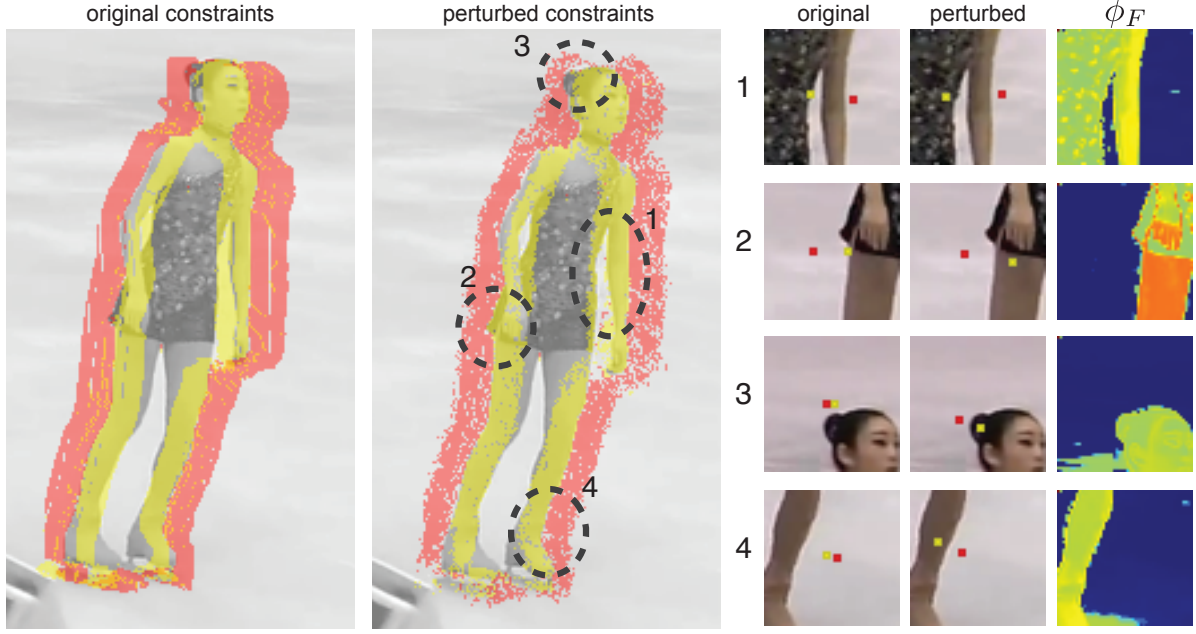


Figure 5: *Constraint perturbation helps clean up poorly estimated constraints. Left: original constraints before perturbation. Middle: improved constraints. Right: several occluder-occluded pixel pairs where perturbation ensures that estimate of the occluder’s position lies on the foreground object (the local foreground probability map is shown under ϕ_F).*

3.2. Local shape classifiers

In some image regions, poor motion estimation or excessive clutter can confuse the temporal integration and computation of segmentation weights ($g_t(x)$ in (11)) as both are based on pixelwise difference—this can make determining object boundaries difficult despite temporal integration. To reliably segment the objects from the background, we incorporate shape information over a local region near the object boundaries. In a fashion similar to [1], we employ a set of overlapping localized shape classifiers to help locally discriminate between foreground objects and the background. Each classifier learns an appearance model and a shape model of the object within a small region, and then propagates this information to the next frame by locally adjusting strength of the foreground prior κ_t .

After the first frame is partitioned, a set of local classifiers is instantiated along object contours to track the boundary of the foreground object o_{fg} within a small window W (31 x 31 pixels). Local shape is determined by the mask of o_{fg} in W , denoted $m(x)$, and the local appearance of o_{fg} is modeled by a GMM (3 modes). A confidence c_{fg} on how well the color model discriminates between o_{fg} and background is computed as

$$c_{fg} = 1 - \frac{\int_W |m(x) - p_c(x)| w_c(x) dx}{\int_W w_c(x) dx}, \quad (23)$$

where p_c is the foreground probability computed from the appearance GMM and $w_c(x) = \exp(-d^2(x)/\sigma_c^2)$, which weights the contribution of each pixel based on $d(x)$ (the distance between x and the foreground-background boundary, computed using a distance transform). σ_c is set to half the window size (for us, 15 pixels) (see (2) in [1] for further details). Next, the local classifier is warped forward by the motion of o_{fg} . Based on c_{fg} , appearance and shape models adaptively combined to provide a prior on which pixels in W are likely to be foreground

$$p_{fg}(x) = c_{fg} p_c(x) + (1 - c_{fg}) L(w_t^{t+1}(x)), \quad (24)$$

where $L(w_t^{t+1}(x))$ is the binary mask $L(x)$ warped into the current frame. Finally, the contributions from each local classifier are combined together and added to $\kappa_t(x)$ in (11).

After the first frame, each classifier is propagated forward until its support contains no foreground objects, at which point it is dropped. New classifiers are instantiated where object boundaries in the current frame’s segmentation are not covered by any

local classifiers. See [1] for further details. Generally, we find that these classifiers improve the segmentation of foreground objects in regions where a strong boundary is not discernible as shown in Fig. 6. In addition, for small objects moving quickly, these classifiers help preserve the object as the foreground prior can sometimes be eroded away as shown in Fig. 7.

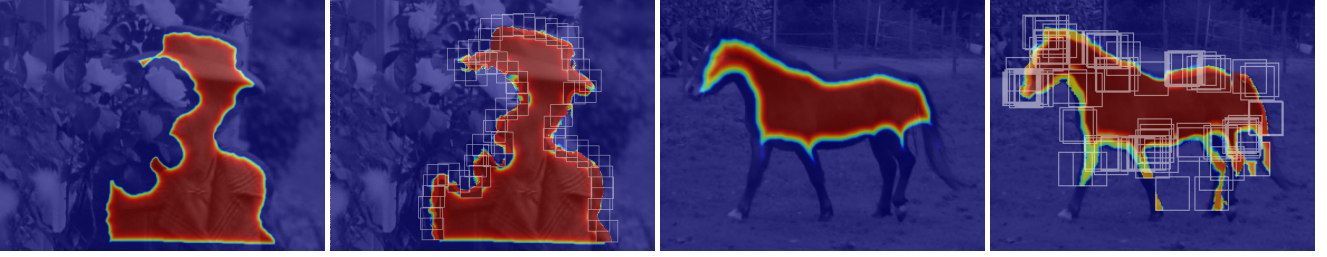


Figure 6: Two examples of the local shape classifiers boosting the foreground prior (κ_t) near object edges, where κ_t is computed without the classifiers on the left and incorporating them (with the classifier windows overlayed in light gray) on the right. Notice how κ_t better captures the shape of the woman's hat behind the bush (left) and the legs of the horse (right).

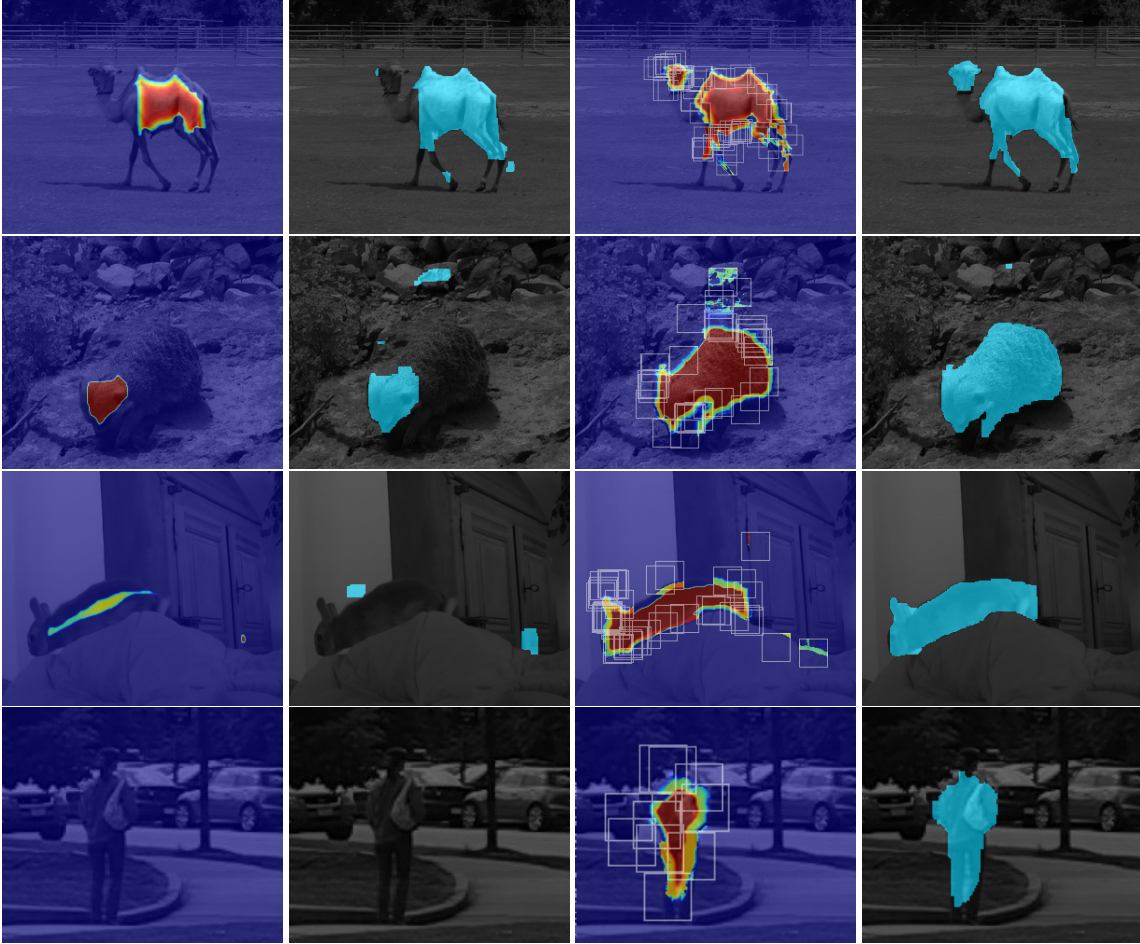


Figure 7: Examples of local shape classifiers preserving small objects when the foreground prior term alone would erode away. From left to right: κ_t without local classifiers, the corresponding layer segmentation, κ_t incorporating the classifiers (with the windows drawn in light gray), and the resulting layers. The camel's head and legs are better captured with the help of local shape classifiers (row 1). Without them, much of the object may go missing (row 2) or even become lost altogether (rows 3–4). These local shape classifiers are beneficial to long-term temporal consistency.

References

- [1] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video snapcut: robust video object cutout using localized classifiers. In *ACM Transactions on Graphics (TOG)*, 2009.
- [2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [3] F. Galasso, S. Naveen, T. J. Cardenas, T. Brox, and B. Schiele. A unified video segmentation benchmark: Annotation, metrics and analysis. In *ICCV*, 2013.