Bayesian-Inspired Space-Time Superpixels

Kent Gauen
Purdue University
gauenk@purdue.edu

Stanley Chan
Purdue University
stanchan@purdue.edu

Abstract

This paper presents Bayesian-inspired Space-Time Superpixels (BIST): a fast, state-of-the-art method to compute space-time superpixels. BIST is a novel extension of a single-image Bayesian method named BASS, and it is inspired by hill-climbing to a local mode of a Dirichlet-Process Gaussian Mixture Model (DP-GMM). The method is only Bayesian-inspired, rather than actually Bayesian, because it includes heuristic modifications to the theoretically correct sampler. Similar to existing methods, BIST can adapt the number of superpixels to an individual frame using split-merge steps. A key novelty is a new temporal coherence term in the split step, which reduces the chance of splitting propagated superpixels. This term enforces temporal coherence in propagated regions, but allows for unconstrained adaptation in disoccluded regions. A hyperparameter determines the strength of this new term, which does not require special tuning to return consistent results across a dataset of videos. The wall-clock runtime of BIST is over twice as fast as BASS and over 30 times faster than the next fastest space-time superpixel method with open-source code.

1. Introduction

Superpixel models [23] represent images as a collection of contiguous, deformably-shaped clusters of perceptually similar pixels. They are inspired by Gestalt principles, which posit humans view images by subconsciously grouping parts into a whole. Recent works show superpixels can be used to improve the quality of deep neural networks (DNNs) for computer vision tasks. [8, 10, 40]. Space-time superpixels extend the single-image concept to video by propagating the same label across multiple frames, with applications in tracking, image registration, and 3D reconstruction. However, the wall-clock runtime of existing space-time superpixel methods can be over one second per frame, which is unacceptably slow. This paper presents Bayesian-inspired Space-Time Superpixels (BIST), a method that achieves state-of-the-art superpixel quality while dramatically reducing the wall-clock runtime.

BIST is a novel extension of the Bayesian Adaptive Superpixel Segmentation (BASS) method and it is distinct from previous space-time work.[1] Previous methods process several frames together, and focus on designing heuristic energy functions to be minimized [3, 12, 13, 25, 26, 38]. In contrast, BIST is designed using heuristic modifications to a theoretically justified sampler of a DP-GMM. BIST proposes no heuristic energy functions to be minimized, which clearly separates this paper from previous work. Ultimately, BIST achieves state-of-the-art benchmark quality and is the fastest available space-time superpixel method with open-source code. **BIST processes images of size 480×320 at over 60 frames-per-second (fps), while the next fastest space-time superpixel method with publicly available code runs at about 2 fps** [3].

BIST consists of three main steps: a shift-and-fill step to move superpixels according to video motion, boundary updates to deform the segmentation boundaries, and split/merge/relabeling steps to adjust the number of superpixels. These steps are common to existing space-time superpixel methods [3, 13, 38]. The novelty of BIST is a new temporal coherence term used within the split step, which adaptively controls the number of split superpixels according to the video dynamics. This term is a heuristic modification to the theoretically correct Hastings ratio used for posterior sampling of a DP-GMM [3, 32]. Hence, the method is Bayesian-inspired rather than actually Bayesian.

The theoretically correct method leads to several hundred split/merge steps when estimating superpixels, and this rapidly replaces old labels with new ones which makes enforcing temporal consistency challenging. The new temporal coherence term works by simply reducing the chance of splitting a propagated superpixel. The strength of this term is controlled by a hyperparameter, and selecting this hyperparameter does not require special tuning to return consistent results across the entire dataset. This term allows BIST to maintain approximately the same number of superpixels per frame as its single-image analogue (BASS) while preserving temporal consistency.

[1] https://gauenk.github.io/bist_website/

	(a)	(b)	(c)	(d)	(e)
GPU Parallelism	✓			✓	✓
Connectivity	✓	✓	✓	✓	
Adaptive Sup. Pix. Num	✓	✓	✓	✓	
Spatial Coherence	✓	✓		✓	
Spatial Covariance	✓			✓	
Temporal Coherence	✓	✓	✓		

Table 1. **Related Works Table.** (a) BIST, (b) TSP [3], (c) Streaming-GBH [38], (d) BASS [32], (e) gSLIC [1]. Connectivity means the superpixels are guaranteed to be contiguous. Spatial coherence means the method encourages more compact superpixel shapes, instead of producing snake-like dendrils. Like BASS, BIST adopts a spatial coherence term that encourages smoother superpixel boundaries and explicitly models the spatial covariance.

2. Related Works

Superpixels form a deformable partition of the input image [23] with a bountiful number of related works. Most work early focused on single-image superpixel estimation [5, 6, 14, 28]. SLIC [1] marks a major milestone for efficient superpixel estimation and there are several closely related methods [16, 24]. Basically, SLIC runs a k-means algorithm over the image pixels in the LAB color space. The main problem with SLIC-like methods is that the estimated superpixels are not contiguous. Later works have been designed to address this problem [12, 17, 29, 33, 39], and fastSCSP [7] presents a method that executes boundary updates in parallel on a GPU to enable incredibly fast, contiguous boundary estimation. BASS [32] builds on this method, adding a Potts term to encourage spatial cohesion and split/merge/relabel steps to adapt the number of superpixels to a particular frame. This method is the launching point for our space-time method, selected for its conceptual clarity, superior superpixel quality, and exceptionally small wall-clock runtime.

Several single-image methods have been extended to space-time, but usually report a lower quality than specialized space-time superpixel methods. GBH [9] is a hierarchical method that can span longer videos, but operates on the entire video sequence at once which leads to excessive memory consumption and a slow runtime. Streaming-GBH [38] presents a streaming alternative to GBH, operating on chunks of frames at a time, but still takes tens of seconds to process each frame. TCS [25] proposed minimizing a heuristic space-time energy function and is much faster than the GBH methods. TSP [3] soon outperformed TCS, and presented their method as an inference problem. However, the boundary updates and split/merge steps of TSP are far less efficient than the single-image BASS method, and, ultimately, this makes TSP more than 30 times slower than BIST. Additionally TSP focuses on refining the optical flow estimate using a forward-and-backward process, while BIST only processes forward in time.

Some recent methods report outperforming TSP [12, 13, 26], but the authors do not provide publicly available code. Even comparing to their closed-source results, BIST is still the fastest space-time superpixel method. CCS [12] reports being 28% faster than TSP, while our method is 15 times faster than CCS. PPM [13] reports an average runtime of 0.20 seconds per frame on SegTrackv2 [15], while BIST averages about 0.015 seconds per frame. Finally, OA-TCS [26] does not report a runtime nor provides code.

2.1. Bayesian Adaptive Superpixel Segmentation (BASS)

We present a detailed summary of Bayesian Adaptive Superpixel Segmentation (BASS) [32] since BIST uses similar concepts and notation. A previous method designed a MCMC algorithm to efficiently sample cluster assignments and parameters from a DP-GMM [2], and BASS applied the theoretical findings to estimating superpixels for a single image. This paper continues in this line of work by extending the single-image superpixels to space-time. BIST is a space-time extension of BASS, using modifications to enforce space-time consistency.

Say an image has a height ($\mathcal{H}$) and width ($\mathcal{W}$) with $\mathcal{HW}$ total pixels. An image pixel is a vector $\boldsymbol{x}_i = [\boldsymbol{a}_i \ \boldsymbol{l}_i]$ of size $F+2$ where $\boldsymbol{a}_i \in \mathbb{R}^F$ are the appearance features and $\boldsymbol{l}_i \in \mathbb{R}^2$ is the 2D spatial location. When working with RGB images ($F=3$), the features are transformed to the *Lab* color space before running the segmentation algorithm.

Superpixel methods often model image pixels as following some variation of a Gaussian Mixture Model (GMM). Loosely speaking, a DP-GMM is a GMM that entertains infinitely many cluster assignments. First for a fixed number of superpixels (S), superpixel assignment at image index i is denoted z_i with $z_i \in \{0,\dots,S-1\}$ and each label has prior probabilities $\{\pi_s\}_{s=0}^{S-1}$. Second, the pixel values are sampled, $\boldsymbol{x}_i \sim p(\boldsymbol{x}|\boldsymbol{\mu}_{z_i}^{\text{app}}, \sigma_{\text{app}}^2, \boldsymbol{\mu}_{z_i}^{\text{shape}}, \Sigma_{z_i}^{\text{shape}})$, which is written as,

$$\begin{aligned} &p(\boldsymbol{x}_i|\boldsymbol{\mu}_{z_i}^{\text{app}}, \sigma_{\text{app}}^2, \boldsymbol{\mu}_{z_i}^{\text{shape}}, \Sigma_{z_i}^{\text{shape}}) \\ &= \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}^{\text{app}}, \sigma_{\text{app}}^2 \boldsymbol{I}_{F\times F}) \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}^{\text{shape}}, \Sigma_{z_i}^{\text{shape}}) \end{aligned} \quad (1)$$

The two Gaussian densities are centered at the mean appearance and shape terms, $\{\boldsymbol{\mu}_s^{\text{app}}, \boldsymbol{\mu}_s^{\text{shape}}\}_{s=1}^S$. The covariance of the appearance term is fixed and isotropic. This controls the balance between regularly shaped superpixels (larger σ_{app}^2) versus deformably-shaped superpixels (smaller σ_{app}^2). The shape covariance matrix allows for ellipsoidal regions. Later, we explore using the anisotropic covariance term to enable better control over the superpixel shape in our space-time method. We abbreviated the appearance and shape parameters $\theta_s^{\text{app}} = \left(\boldsymbol{\mu}_s^{\text{app}}, \sigma_{\text{app}}^2 \boldsymbol{I}_{F\times F}\right)$ and

Algorithm 1 Bayesian-Inspired Space-Time Superpixels (BIST)

```
Require: Frame ($\boldsymbol{x}^{(t)}$), Flow ($\boldsymbol{f}^{(t-1)}$), Superpixels ($\boldsymbol{z}^{(t-1)}$), Hyperparameters ($\varphi$)
 1: $\boldsymbol{z}, \tilde{\boldsymbol{z}} \leftarrow$ Shift-and-Fill($\boldsymbol{z}^{(t-1)}, \boldsymbol{f}^{(t-1)}$)    ▷ Sec. 3.1
 2: $N \leftarrow \{4, 8, 12\}$ based on $\tilde{\boldsymbol{z}}$    ▷ Sec. 3.1
 3: for $i = 1$ to N do
 4:   if i mod 4 = 1 then
 5:     $\boldsymbol{z} \leftarrow$ Split($\boldsymbol{x}^{(t)}, \boldsymbol{z}, \tilde{\boldsymbol{z}}, \varphi$)    ▷ Sec. 3.3
 6:     $\boldsymbol{z} \leftarrow$ Boundary Updates($\boldsymbol{x}^{(t)}, \boldsymbol{z}, \varphi$)    ▷ Sec. 3.2
 7:   else if i mod 4 = 3 then
 8:     $\boldsymbol{z} \leftarrow$ Relabel($\boldsymbol{x}^{(t)}, \boldsymbol{z}, \varphi$)    ▷ Sec. 3.4
 9:     $\boldsymbol{z} \leftarrow$ Merge($\boldsymbol{x}^{(t)}, \boldsymbol{z}, \varphi$)    ▷ Sec. 3.4
10:     $\boldsymbol{z} \leftarrow$ Boundary Updates($\boldsymbol{x}^{(t)}, \boldsymbol{z}, \varphi$)
11:   else
12:     $\boldsymbol{z} \leftarrow$ Boundary Updates($\boldsymbol{x}^{(t)}, \boldsymbol{z}, \varphi$)
13:   end if
14: end for
15: return $\boldsymbol{z}, \phi$
```

Algorithm 2 Applying BIST to a Video

```
Require: Video ($\{\boldsymbol{x}^{(t)}\}_{t=1}^{T}$), Flows ($\{\boldsymbol{f}^{(t)}\}_{t=1}^{T}$), Hyperparameters ($\varphi$)
 1: $\boldsymbol{z}^{(1)} \leftarrow$ BASS($\boldsymbol{x}^{(1)}, \varphi$)    ▷ [32]
 2: for $t = 2$ to $T$ do
 3:   $\boldsymbol{z}^{(t)} \leftarrow$ BIST($\boldsymbol{x}^{(t)}, \boldsymbol{f}^{(t-1)}, \boldsymbol{z}^{(t-1)}, \varphi$)    ▷ Alg. 1
 4: end for
 5: return $\{\boldsymbol{z}^{(t)}\}_{t=1}^{T}$
```

$\theta_s^{\text{shape}} = \left(\boldsymbol{\mu}_s^{\text{shape}}, \Sigma_s^{\text{shape}}\right)$. BASS also includes a Potts term to encourage regularly shaped superpixels, which we use and explain in Section 3.2.

BASS superpixel estimation consists of two main steps. The first step is boundary updates, which iterates between updating the boundary of the superpixel segmentation and re-estimating superpixel parameters $(\theta_s^{\text{app}}, \theta_s^{\text{shape}})_{s=1}^{S}$. Boundary updates reclassify superpixels that are adjacent to two or more distinct superpixel labels, assigning them to match the class of one of their neighbors. The segmentation update can be done in parallel on a GPU [7, 24]. Superpixel parameters are updated with their maximum likelihood estimates. The exception is the spatial covariance term, which uses the posterior mode computed using a prior shape, $\lambda \boldsymbol{I}_{2\times 2}$. The hyperparameter λ controls the size of the superpixels.

3. Bayesian-Inspired Space-Time Superpixels (BIST)

BIST is a method to estimate space-time superpixels, and it is theoretically inspired by hill-climbing to the posterior mode of a DP-GMM. The BIST method is presented in Algorithm 1, and consists of three major parts. First in the Shift-and-Fill step, superpixels are propagated from frame t to frame $t+1$ according to the input optical flow. Propagating superpixels from the previous frame acts as a good initialization to encourage temporally coherent labels. Second, these superpixels are refined via boundary updates, similar to recent works [7, 32]. We explored using informative priors in this step and found a consistent but marginal improvement in superpixel quality. Third, superpixels can be split, merged, and/or relabeled to control the total number of supeprixels used to explain the image. Splits principally explain disocclusion, merging removes unnecessary superpixels, and relabeling allows for correcting erroneous tracking and enables tracking objects through occlusion. BIST is designed to be the space-time extension of BASS [32], as both the boundary updates and split-merge steps of BIST are based on BASS.

Inputs to BIST include a video $\{\boldsymbol{x}^{(t)}\}_{t=1}^{T}$ and a dense flow field $\{\boldsymbol{f}^{(t)}\}_{t=1}^{T}$ where each frame has sizes $\mathcal{HW} \times F$ and $\mathcal{HW} \times 2$, respectively. The last input is the BIST hyperparameters ($\varphi = \{\lambda, \sigma^2_{\text{app}}, \beta, \alpha, \gamma, \varepsilon_{\text{re-id}}, \varepsilon_{\text{new}}\}$) which include the initial superpixel size (λ), appearance variance (σ^2_{app}), the strength of the Potts term (β), the merge Hastings hyperparameter (α), the strength of the split-step temporal coherence term (γ), and the parameters for re-identification and new label creation ($\varepsilon_{\text{re-id}}, \varepsilon_{\text{new}}$). Algorithm 2 demonstrates how BIST is applied to a video; BASS is used to estimate superpixels for frame 1, and subsequent frames apply BIST.

3.1. Shift-and-Fill

The first step of the space-time superpixel method is propagating the superpixels from frame t to frame $t+1$. Motion between two frames of a video is commonly modeled as the shifting of pixel values to new image coordinates [19]. Similar to other space-time superpixel methods [3, 13], this paper models motion as shifting superpixels rather than pixels. Shifting creates regions without a superpixel label (holes), which are iteratively filled with a watershed-like algorithm.

***Shift* Superpixels.** Superpixels from one frame are propagated to the next by shifting each superpixel as a single unit according to an optimal flow estimate. Unlike the related works which estimate flow as part of their methods, this paper uses a high-quality dense optical flow from a recent deep neural network method [22, 31]. Finding a good initial estimate is important since the subsequent steps will make mostly local changes, but pixel-perfect refinement is

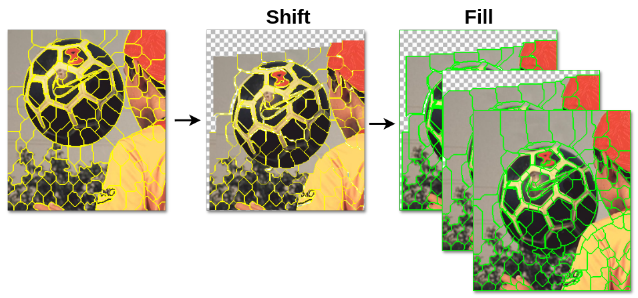

Figure 1. **Shift and Fill.** The initialization of the propagated superpixels is important for temporal consistency since later steps only refine this initial estimate. Superpixel labels from frame t are shifted to frame $t+1$ according to the average optical flow of the superpixel. The result is depicted in the center image, which shifts the segmentations across the image like puzzle pieces moving across a board. This leads to overlapping regions and holes. For overlapping regions, the smallest superpixels are displayed on top. The holes are iteratively filled to ensure the missing superpixel labels are initialized to contiguous regions.

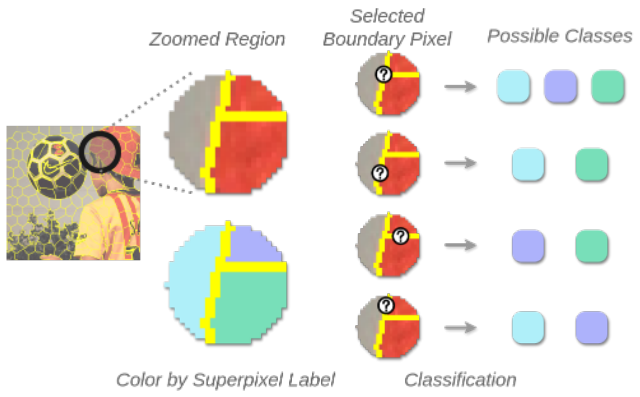

Figure 2. **Boundary Updates.** Boundary updates reclassify superpixels that are adjacent to two or more distinct superpixel labels, assigning them to match the class of one of their neighbors. This is similar to running k-means clustering updates but restricted to the subset of pixels that run along the edges of each superpixel.

not necessary since subsequent boundary updates will adjust the initial superpixel positions.

Figure 1 illustrates the shifting. First, a dense optical flow is estimated using any method, say RAFT or SpyNet [22, 31]. These dense optical flows are then converted into a single flow vector for each superpixel by averaging the flow across all pixels within a superpixel. These average flow fields shift the superpixel labels to the new location.

***Fill*-ing in Missing Superpixels.** Shifting superpixels can yield overlapping superpixels and regions without a superpixel label, aka holes. Overlapping superpixels are handled with a deterministic z-axis ordering. Similar to recent work [4], the smallest superpixel is selected when two or more superpixels overlap. Smaller superpixels are more likely to be erroneously removed during the superpixel shift, so placing them in front gives smaller superpixels the chance to survive in the next frame. This can be done efficiently using a customized atomic operator that records the associated label while executing an atomic min.

Holes are imputed with a watershed-like algorithm [34] that iteratively grows valid superpixel regions. Each missing point bordering at least one existing point is assigned to the most similar superpixel label (using mean appearance). In many cases, the proposed watershed algorithm produces contiguous superpixels. Still, disconnected superpixels can be created if one superpixel intersects another after shifting. In these few cases, we simply split the superpixel, assigning the original label to the largest chunk and new labels to the smaller chunks.

As presented in Algorithm 1, the number of iterations actually adapts to the motion of the underlying scene. If more than 20% of the superpixels after the shift step are invalid, then BIST runs for 12 iterations. If less than 1% of the superpixels are invalid, then BIST runs for 4 iterations. Otherwise (the common case), BIST runs for 8 iterations. On average, BIST runs for 8.113 iterations and we report no sequence with an average of over 10 iterations. In comparison, the number of iterations BASS runs is equal to the initial superpixels size, which we set to 20. In Section 4.2, we show how this will decrease the wall-clock runtime of BIST compared to BASS.

3.2. Boundary Updates

Boundary updates are what give superpixels their distinctive deformable shapes, and this step can be updated in parallel on a GPU [32]. For each pixel at the border of two or more superpixels (aka the boundary), the superpixel label is re-classified to one of its neighbors. These updates are similar to cluster assignments in k-means [18], following an Expectation-Maximization (EM)-like algorithm where parameters are estimated based on label assignments, followed by reassigning labels using these updated parameters.

While the label re-classification and parameter estimates are two distinct steps, this paper combines them for a succinct presentation. Specifically, we say the boundary update step starts with updating the model parameters, which is a straight-forward reduction operation on a GPU. The parameter update step is nothing special, so we regulate these details to Supplemental Section 8.1.

Once the parameters are computed, the classification step re-assigns each pixel on the boundary to their most likely neighboring cluster. The conditional probability over superpixel labels for a single point is given as,

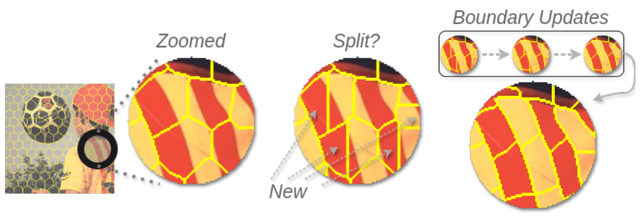

Figure 3. **Split via Deferred Sampling.** Split steps allow the number of superpixels to be adapted to the current image. The split step in BASS uses a method called *deferred sampling*, which is motivated by posterior sampling from a DP-GMM. First, superpixels are split through their center (horizontally or vertically). Second, the superpixel segmentation is updated via boundary updates. This procedure yields theoretically correct samples.

$$p(z_i|\boldsymbol{x}_i, \boldsymbol{l}_i, \theta_{z_i}, \boldsymbol{z}_{\not i}) \propto \pi_{z_i}\mathcal{N}(\boldsymbol{x}_i; \theta^{z_i}_{\text{app}})\mathcal{N}(\boldsymbol{l}_i; \theta^{z_i}_{\text{shape}})p(z_i|\boldsymbol{z}_{\not i}) \tag{2}$$

where $\theta_{\text{app}}, \theta_{\text{shape}}$ are parameters of the Gaussians described in Section 2.1. The right-most density is the Potts term, and acts like an indicator function to only permit contiguous labeling and favoring cohesive shapes. The density is written as,

$$p(z_i|\boldsymbol{z}_{\not i}) = \mathbb{1}_{\text{valid}}(\boldsymbol{z}) \exp\left\{-\beta \sum_{i\sim i'} \mathbb{1}_{z_i \neq z_{i'}}(\boldsymbol{z})\right\} \tag{3}$$

The likelihood of the pixel is computed for each neighboring superpixel, and the label is updated to the most likely class. While there are a few technical details associated with the parallel updates, the result is a fast method to update superpixel boundaries. By only updating a particular subset of points (called *simple points*) in parallel, boundary updates are guaranteed to form contiguous regions [7]. We choose $\beta = 10$ since it produces visually appealing results.

3.3. A Temporally Coherent Split Step

Many superpixel methods adaptively control the number of superpixels according to the image content [3, 9, 32]. Increasing the number of superpixels is done by splitting existing superpixels and decreasing the number of superpixels is done by merging two neighboring superpixels together. In the single-image case and/or when superpixels are new to a frame, split-merge steps are used as compliments in the optimization process, but in the space-time setting the two steps are conceptually distinct. Split steps assign superpixels to explain disoccluded regions in an adjacent frame, while merge steps merely reduce the total number of superpixels with no physical meaning. In other words, adapting the number of superpixels to an individual frame must be balanced with propagating superpixels across time. This subsection presents details on the more complicated and conceptually relevant split step.

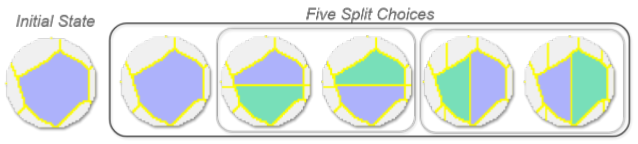

Figure 4. **Classifying the Split.** The split step requires classifying each superpixel as not-split, split horizontally, or split vertically. Determining which half of the split superpixel is propagated (purple) or new (green) yields five classes. BIST introduces a temporal term to BASS's classifier.

Deferred Sampling. At a high-level, split steps are complicated because identifying a likely split is identical to fixed-K superpixel estimation, which gives a recursive flavor to the problem. TSP [3]'s solution is to run k-means within each superpixel, but this requires a significant computational overhead. BASS [32] uses a less expensive method inspired by *deferred sampling* within the sampler of a DP-GMM [2]. The idea is to vertically or horizontally split a superpixel if a function of its parameters (the Hastings ratio) exceeds some threshold. Once split, the entire segmentation is updated for a few iterations without further splits/merges. This procedure is theoretically justified to produce correct samples [2]. Figure 3 illustrates three superpixels split vertically and one superpixel split horizontally. Clearly, the initial segmentation is qualitatively bad. However, via deferred sampling, one can continue executing boundary updates across the entire image to ensure the resulting segmentation is a likely sample.

Rapid Cycling. Naïvely using the DP-GMM Hastings ratio yields several hundred splits and merges (rapid cycling) during superpixel estimation, which is problematic for temporal coherence. Rapid cycling replaces superpixels of one label with superpixels of a different label, yet both labels cover an almost identical region. This rapid cycling can be slow, and in the space-time case, it can be detrimental to temporal coherence. Therefore, a new classification criterion should be used to balance adaptation to the current image and temporal coherence across the video.

The Split Step. To reduce the number of unnecessary splits and encourage temporal coherence, BIST introduces a new term in the Hastings ratio of the split step. The term consists of a hyperparameter (γ) to control its strength and the propagated label density of each superpixel, $\{\rho_s\}_{s=1}^{S}$. This density is the percent of valid labels within each superpixel after the Shift-and-Fill step. Essentially, the new term makes splitting more difficult for propagated superpixels.

A split step starts by temporarily cutting each superpixel, and the Hastings ratio determines if this cut is kept using a ratio of marginal likelihood terms. For the sake of presentation, lets assume a superpixel is cut into a left and right half. Concretely, this means half of a superpixel's labels are temporarily assigned to a new value and summary statistics are computed by simply executing a reduction on a GPU. Let $f(\boldsymbol{x}^s)$, $f(\boldsymbol{x}^{s_l})$, and $f(\boldsymbol{x}^{s_r})$ denote the marginal likelihoods associated with the whole superpixel s and each split half. While the theoretically the marginal likelihood should include both the appearance and shape terms, BASS found using appearance parameters gave nice results and we report a similar finding. Since the appearance parameters follow a Normal-Inverse Gamma distribution, the marginal likelihood is a simple function of the pixel values.

Let's denote the prior as $\mathrm{NIG}(\mu_s^{\text{app}}, \sigma_s^{2\ \text{app}}; m_s, \kappa_s, a_s, b_s)$ where the parameters are fixed as follows: $m_s = 0$, $\kappa_s = 0$, $a_s = 10^4$ and $b_s = \cancel{\sigma^{2\ \text{app}}(\cdot 10^4 - 1)} = 2\,\sigma^{\text{app}} \cdot 10^4$. While the first equality for b_s is theoretically correct, the heuristic change improves results. And while $\kappa_s = 0$ is improper, the corresponding terms in the marginal likelihood are simply dropped. Let $(n_s, v_s, \boldsymbol{a}_s', \boldsymbol{a}_s'') = \sum_{i:z_i=s}(1, \tilde{z}_i \geq 0, \boldsymbol{a}_i, \boldsymbol{a}_i \odot \boldsymbol{a}_i)$ be summary statistics of superpixel s, where $\odot$ refers to element-wise multiplication and $\rho_s = \frac{v_s}{n_s}$ is the density of valid superpixel labels after the shift step. We refer the reader a reference for a complete derivation of the marginal likelihood (see Eq 55) [20], and write down the solution below,

$$f(\boldsymbol{x}^s) = \int_\Omega \prod_{i \in \{j : z_j = s\}} p(\boldsymbol{x}_i | \theta^{\text{app}}) p(\theta^{\text{app}})\ d\theta^{\text{app}}$$
$$= \frac{\cancel{|\kappa + n_s|}^{\frac{1}{2}} b^a \Gamma(a + n_s/2)}{\cancel{|\kappa|}^{\frac{1}{2}} \hat{b}^{(a+n_s/2)} \Gamma(a) \pi^{\frac{n_s}{2}} 2^{n_s}} \tag{4}$$

where $\hat{b} = b_s + \frac{1}{2}\left(\boldsymbol{a}_s'' - (\boldsymbol{a}_s')^2/n_s\right)$. The red strike-through indicated heuristically dropped terms. Notice that although the appearance variance is a fixed hyperparameter, it is treated as an unknown quantity when deriving the marginal likelihood. The Hastings ratio is a ratio of these marginal likelihoods as follows,

$$\cancel{\frac{\alpha\Gamma(n_{s_l})\Gamma(n_{s_r})}{\Gamma(n_s)}} \frac{f(\boldsymbol{x}^{s_r}) f(\boldsymbol{x}^{s_r})}{f(\boldsymbol{x}^s)} > e^{-2} e^{\gamma \rho_s} \tag{5}$$

where the red strike-through indicates heuristically removed/included in BASS and the blue quantity is the novel temporal coherence term in BIST. Since $\gamma \geq 0$ is a hyperparameter, this new term makes splitting less likely if the superpixel labels are propagated ($\rho_s \approx 1$), since the threshold to accept a split increases. This term "turns-off" if the superpixel labels are missing after the shift step ($\rho_s \approx 0$), and resolves back to the criteria used in the single-image case. Once a split is accepted, the half with fewer missing labels is given the propagated label. The strength (γ) of the temporal coherence term can be modified to reduce the number of new superpixels without significantly degrading benchmark quality.

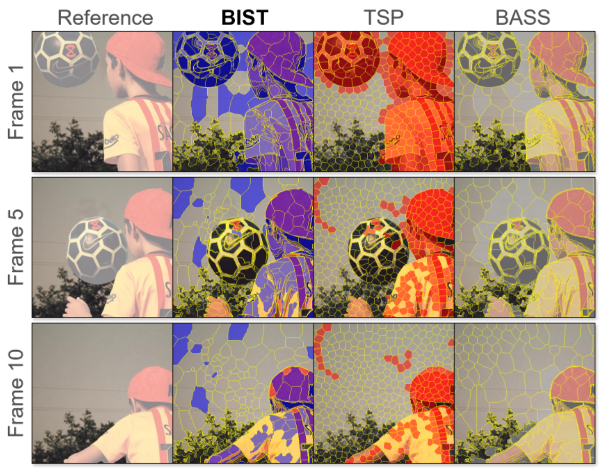

Figure 5. **Qualitatively Comparing Space-time Superpixels.** Similar to BASS, BIST superpixels fit oddly shaped regions and adaptively estimate the number of superpixels. On average, BIST and TSP have similar temporal coherence.

3.4. Merges and Relabeling

Merges. When merging, at most one superpixel can be propagated from a previous frame. This is justified by our model that superpixels shift to different pixel locations, so merging two conditioned superpixels has no physical meaning. Otherwise, this step exactly matches the BASS update [32] and is detailed in Supplemental Section 8.3.

Relabeling. BIST uses a standard two-step relabeling approach similar to previous work [3, 13]: re-identification to handle occlusions by connecting active superpixels to previously seen inactive ones, and new label creation to maintain appearance consistency by creating new labels when substantial changes to the appearance and shape means occur. This mechanism is particularly effective for handling optical flow inaccuracies, as improperly propagated superpixels appropriately relabeled as new superpixels. The full details of this step is provided in the Supplemental Section 8.4.

4. Experiments

4.1. Space-Time Superpixel Quality

Setup. To quantitatively evaluate the superpixels, this paper uses LIBSVX [36, 37] and the Lib-Stutz [30]. The single-image superpixel methods for comparison include BASS [32], ERS [17], ETPS [39], SEEDS [33], and SLIC [1]. The space-time superpixel methods for comparison include TSP [3] and Steaming-GBH [38]. One selected dataset is the SegTrackv2 [15], which is a collection

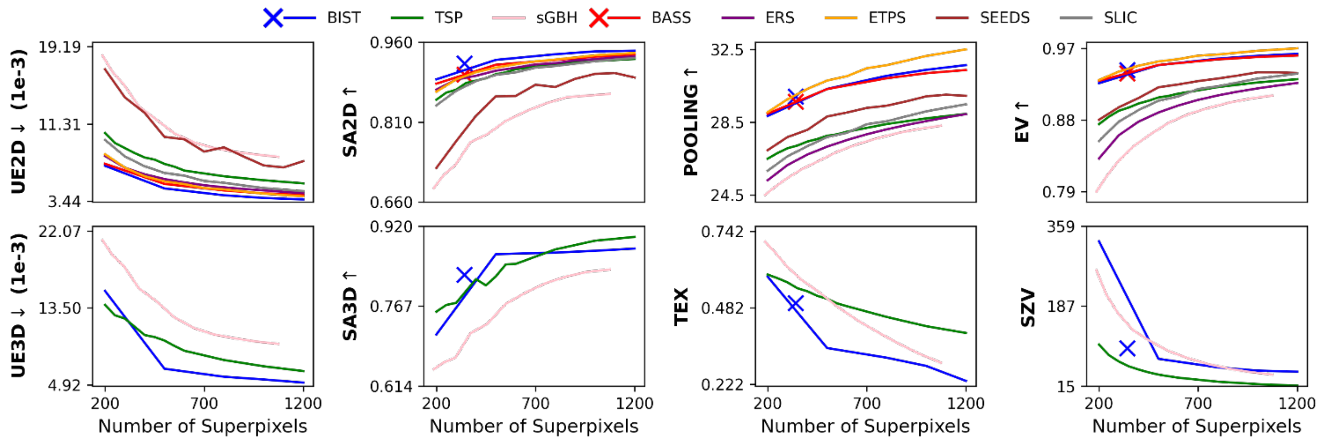

Figure 6. **Quantitative Superpixel Comparison on SegTrackv2.** This figure quantitatively compares BIST against existing superpixel methods on standard benchmarks. BIST demonstrates exceptional results on the superpixel metrics but suffers from a shorter temporal extend (TEX) than TSP, meaning the superpixels stay alive for fewer frames. This is partially due to the small lifespan of smaller superpixels (see Section 7). As BIST and BASS adaptively estimate the number of superpixels to each scene, their default results are marked with a x.

of about 1000 frames from 14 sequence categories with resolution about 320×480. Another dataset is the DAVIS 2017 validation dataset [21], which consists of 30 sequences with about 50 - 85 frames per sequence with resolution about 480×960.

Quantitative Benchmarks. Following related works, we assess superpixel quality on standard superpixel benchmarks: Segmentation Accuracy (SA2D/SA3D), Undersegmentation Error (UE2D/UE3D), and Expected Variance (EV). The first two metrics assess the adherence of the superpixels to a groundtruth segmentation label. SA (also known as Achieve Segmentation Accuracy) reports the best possible segmentation computed using the estimated superpixels. UE quantifies the leakage of superpixels across ground-truth segmentation. EV reports the variation of pixel values explained by superpixel assignments. This is related to also the superpixel pooling quality, which compares the superpixel means with the original image [8, 10]. Quantitative space-time benchmarks include temporal extent (TEX) and size variation (SZV). TEX reports the average lifetime of a superpixel and SZV reports the variance of the superpixel's size across time. We direct the reader to the following references for a detailed catalog of these metrics [30, 36].

Results. Figures 5 qualitatively compares BIST with TSP [3]. The graphic illustrates that BIST superpixels are qualitatively similar to BASS and can have a similar temporal extent to TSP when objects are moving. Figure 6 quantitatively compares the superpixels on standard benchmarks. BIST superpixels report excellent single-image and space-time superpixel benchmarks. The tabular representation of this information for BIST, BASS, and TSP is presented in Table 5 (Sec 10). BIST exhibits a slightly shorter temporal extent than TSP, and this related to BIST's irregular superpixel shapes. This limitation is further discussed in Supplemental Section 7.

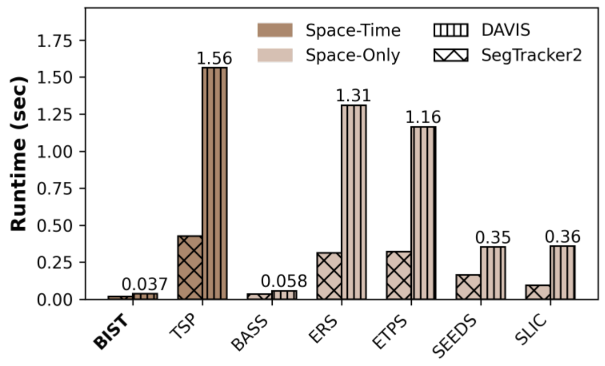

Figure 7. **Comparing Superpixel Runtimes.** This graphic compares the per-frame runtime of space and space-time superpixels on the SegTrackv2 [15] dataset with image resolution about 320×480 and DAVIS [21] dataset with image resolution about 480×960. The wall-clock runtime is reported in seconds. Steaming-GBH [38] is not plotted because the method is too slow. The numbers on top of the DAVIS bars is the median runtime.

Both BIST and BASS automatically determine the number of superpixels, so fixing a target number of superpixels for each method presents a challenge. In this paper, we designed a small subroutine to ensures that both methods are within 5% of the indicated number of superpixels. See Supplemental Section 8.5 for more details.

4.2. Computation

Figure 7 compares the wall-clock runtime of competing superpixel methods. BIST can be more than 30 times faster

# Spix	Type	TEX (%)	SZV	Pool (dB)↑	SA-2D↑	SA-3D↑	UE-2D↓	UE-3D↓
1300	$\gamma\rho_s$	30.6 ± 3.0	111.4 ± 7.4	27.91 ± 0.38	0.919 ± 0.017	0.854 ± 0.021	4.75 ± 0.41	8.05 ± 0.74
	α	28.8 ± 2.8	122.8 ± 8.0	27.99 ± 0.38	0.919 ± 0.017	0.853 ± 0.021	4.83 ± 0.42	8.16 ± 0.76
1180	$\gamma\rho_s$	30.7 ± 3.0	124.9 ± 8.5	27.79 ± 0.38	0.917 ± 0.017	0.851 ± 0.020	4.98 ± 0.44	8.40 ± 0.78
	α	29.2 ± 2.8	139.5 ± 8.6	27.80 ± 0.38	0.915 ± 0.018	0.848 ± 0.022	5.13 ± 0.45	8.65 ± 0.79
1065	$\gamma\rho_s$	31.9 ± 3.2	136.4 ± 9.2	27.58 ± 0.38	0.912 ± 0.019	0.843 ± 0.021	5.18 ± 0.45	8.74 ± 0.79
	α	28.9 ± 2.8	161.4 ± 9.1	27.50 ± 0.38	0.909 ± 0.018	0.842 ± 0.021	5.51 ± 0.48	9.31 ± 0.82

Table 2. **Quantitative Impact of the Temporally Coherent Split Step.** This table compares the benchmark results of BIST superpixel where the number of superpixels was controlled using either the temporally coherent split step ($\gamma\rho_s$) or adjusting the single-image Hastings merge-step hyperparameter (α). The proposed split step term yields higher-quality results. The scale for UE-2D and UE-3D is 10^{-3}. Results are reported with standard errors divided by $\sqrt{30}$ videos.

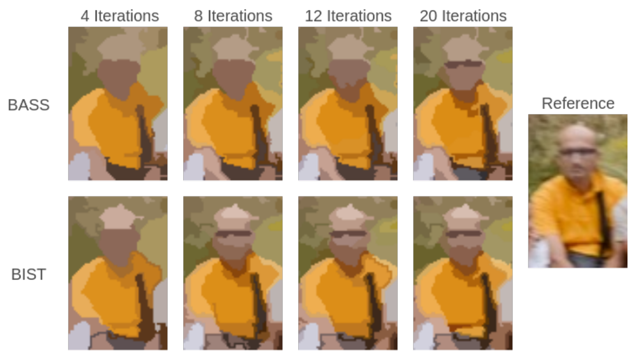

Figure 8. **BIST Requires Fewer Iterations than BASS.** This image depicts the superpixel-pooled image when stopping BIST and BASS at the indicated iteration. The visualization shows that BIST requires fewer iterations than BASS. For BIST, the sunglasses appear at iteration 8, while they appear only at iteration 20 in BASS.

than the next fastest space-time superpixel method with open-source code (TSP [3]) and it can be more than twice as fast as BASS. In fact, BIST is the fastest superpixel method we ran. However, we note that the single-image methods were executed from within a docker which does introduce a slight overhead. The improved runtime from TSP to BIST is due to the efficient, novel split step in this paper and the efficient boundary updates from related works [7]. The improved runtime of BIST over BASS [32] is due to the fact that fewer iterates are necessary in the space-time case. The number of iterations BIST runs adapts changes depending on the video motion, while the number of iterations BASS runs is equal to the initial superpixels size, which we set to 20. To highlight the importance of running BASS for the full 20 iterations, Figure 8 compares the superpixel-pooled images from BASS and BIST when run for 4, 12, and 20 iterations. The sunglasses appear earlier for BIST compared to BASS, indicating BIST requires fewer iterations.

4.3. Temporally Coherent Split Step

A major novelty of BIST is the temporally coherent split step term. The term restricts the creation of new superpixels in regions that have been propagated from a previous frame. Logically, this is appealing because the space-time split step resolves back to the singe-image split step in regions that are not propagated across frames. That is, the new term "turns-off" in regions are do not requires space-time consistency. However, the average number of superpixels per frame could conceivably be controlled using the merge step α hyperparameter. Conceptually, this is undesirable because the α term is given two simultaneous roles; enforcing space-time consistency and single-image adaptation. So as a practical matter, we investigate the experimental value of using the proposed $\gamma\rho_s$ term versus using only the single-image α term.

Table 2 compares the results of these two approaches across three different average superpixel counts. Since BIST automatically estimates the appropriate number of superpixels, comparisons based solely on hyperparameter choices would be misleading. Instead, we match results based on the average number of superpixels to ensure a fair comparison. This explains the atypical spacing between the number of superpixels in Table 2. For this experiment, we executed a grid of BIST methods with various hyperparameters and matched comparable configurations. All matched results are within 2% of the reported superpixel counts. Table 6 in the Supplemental Section 10 reports the hyperparameters for each result. Our findings demonstrate that the proposed temporally coherent split term yields significantly higher-quality superpixels than using the space-only merge hyperparameter.

5. Conclusion

BIST is a Bayesian-inspired space-time superpixel method that is faster and higher-quality than previous methods. BIST is designed by making heuristic modifications of a DP-GMM sampler, rather than derived from energy functions. The core novelty is a temporally coherent split step term to encourage temporal coherence. The real-time wall-clock runtime of BIST makes it feasible to use temporally coherent superpixels within deep neural network pipelines.

6. Acknowledgments

The work is supported, in part, by the US National Science Foundation under the awards 2133032, 2134209, and a Semiconductor Research Corporation JUMP 2.0 Center grant.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. 2, 6

[2] Jason Chang and John W Fisher III. Parallel sampling of dp mixture models using sub-cluster splits. *Advances in Neural Information Processing Systems*, 26, 2013. 2, 5

[3] Jason Chang, Donglai Wei, and John W Fisher. A video representation using temporal superpixels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2051–2058, 2013. 1, 2, 3, 5, 6, 7, 8

[4] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Alexander Schwing, and Joon-Young Lee. Tracking anything with decoupled video segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1316–1326, 2023. 4

[5] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002. 2

[6] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59:167–181, 2004. 2

[7] Oren Freifeld, Yixin Li, and John W Fisher. A fast method for inferring high-quality simply-connected superpixels. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 2184–2188. IEEE, 2015. 2, 3, 5, 8

[8] Kent Gauen and Stanely Chan. Soft superpixel neighborhood attention. *Advances in neural information processing systems*, 38, 2024. 1, 7, 3

[9] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Efficient hierarchical graph-based video segmentation. In *2010 ieee computer society conference on computer vision and pattern recognition*, pages 2141–2148. IEEE, 2010. 2, 5

[10] Varun Jampani, Deqing Sun, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Superpixel sampling networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 352–368, 2018. 1, 7

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[12] Se-Ho Lee, Won-Dong Jang, and Chang-Su Kim. Contour-constrained superpixels for image and video processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2443–2451, 2017. 1, 2

[13] Se-Ho Lee, Won-Dong Jang, and Chang-Su Kim. Temporal superpixels based on proximity-weighted patch matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3610–3618, 2017. 1, 2, 3, 6

[14] Alex Levinshtein, Adrian Stere, Kiriakos N Kutulakos, David J Fleet, Sven J Dickinson, and Kaleem Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2290–2297, 2009. 2

[15] Fuxin Li, Taeyoung Kim, Ahmad Humayun, David Tsai, and James M Rehg. Video segmentation by tracking many figure-ground segments. In *Proceedings of the IEEE international conference on computer vision*, pages 2192–2199, 2013. 2, 6, 7

[16] Zhengqin Li and Jiansheng Chen. Superpixel segmentation using linear spectral clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1356–1363, 2015. 2

[17] Ming-Yu Liu, Oncel Tuzel, Srikumar Ramalingam, and Rama Chellappa. Entropy rate superpixel segmentation. In *CVPR 2011*, pages 2097–2104. IEEE, 2011. 2, 6

[18] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982. 4

[19] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI'81: 7th international joint conference on Artificial intelligence*, pages 674–679, 1981. 3

[20] Kevin P Murphy. Conjugate bayesian analysis of the gaussian distribution, 2007. 6, 2

[21] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 7, 4

[22] Anurag Ranjan and Michael J. Black. Optical flow estimation using a spatial pyramid network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3, 4, 6

[23] Ren and Malik. Learning a classification model for segmentation. In *Proceedings ninth IEEE international conference on computer vision*, pages 10–17. IEEE, 2003. 1, 2

[24] Carl Yuheng Ren and Ian Reid. gslic: a real-time implementation of slic superpixel segmentation. *University of Oxford, Department of Engineering, Technical Report*, pages 1–6, 2011. 2, 3

[25] Matthias Reso, Jorn Jachalsky, Bodo Rosenhahn, and Jorn Ostermann. Temporally consistent superpixels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 385–392, 2013. 1, 2

[26] Matthias Reso, Jörn Jachalsky, Bodo Rosenhahn, and Jörn Ostermann. Occlusion-aware method for temporally consistent superpixels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(6):1441–1454, 2018. 1, 2

[27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015. 4

[28] Shi. Multiclass spectral clustering. In *Proceedings ninth IEEE international conference on computer vision*, pages 313–319. IEEE, 2003. 2

[29] David Stutz, Alexander Hermans, and Bastian Leibe. Superpixel segmentation using depth information. *RWTH Aachen University, Aachen, Germany*, 4, 2014. 2

[30] David Stutz, Alexander Hermans, and Bastian Leibe. Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding*, 166:1–27, 2018. 6, 7

[31] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*, pages 402–419. Springer, 2020. 3, 4, 6

[32] Roy Uziel, Meitar Ronen, and Oren Freifeld. Bayesian adaptive superpixel segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8470–8479, 2019. 1, 2, 3, 4, 5, 6, 8

[33] Michael Van den Bergh, Xavier Boix, Gemma Roig, and Luc Van Gool. Seeds: Superpixels extracted via energy-driven sampling. *International Journal of Computer Vision*, 111: 298–314, 2015. 2, 6

[34] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13 (06):583–598, 1991. 4

[35] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 4

[36] Chenliang Xu and Jason J Corso. Evaluation of super-voxel methods for early video processing. In *2012 IEEE conference on computer vision and pattern recognition*, pages 1202–1209. IEEE, 2012. 6, 7

[37] Chenliang Xu and Jason J Corso. Libsvx: A supervoxel library and benchmark for early video processing. *International Journal of Computer Vision*, 119:272–290, 2016. 6

[38] Chenliang Xu, Caiming Xiong, and Jason J Corso. Streaming hierarchical video segmentation. In *European Conference on Computer Vision*, pages 626–639. Springer, 2012. 1, 2, 6, 7

[39] Jian Yao, Marko Boben, Sanja Fidler, and Raquel Urtasun. Real-time coarse-to-fine topologically preserving segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2947–2955, 2015. 2, 6

[40] Aiping Zhang, Wenqi Ren, Yi Liu, and Xiaochun Cao. Lightweight image super-resolution with superpixel token interaction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12728–12737, 2023. 1