Spatula: Efficient Cross-Camera Video Analytics on Large Camera Networks

Samvit Jain, Xun Zhang, Yuhao Zhou, Ganesh Ananthanarayanan, Junchen Jiang, Yuanchao Shu,
Paramvir Bahl, Joseph Gonzalez

Authors

ACM/IEEE Symposium on Edge Computing (SEC 2020), November 2020

- Samvit Jain - UC Berkeley, Masters
- Xun Zhang - University of Chicago, PhD Student
- Yuhao Zhou - University of Chicago, PhD Student
- Ganesh Ananthanarayanan - Microsoft Research
- Junchen Jiang - University of Chicago, Assistant Professor
- Yuanchao Shu - Microsoft Research
- Paramvir Bahl - Microsoft Research
- Joseph Gonzalez - UC Berkeley, EECS Assistant Professor

Overview: Video Analytics

- Applications
  - Localize suspects after security incidents
  - Vehicle tracking
  - Identifying shoppers
- Large data sources
- Cross-camera analytics

Video Analytics Pipelines

- **Object detection module**: extracts and classifies objects of interest in each video frame
- **Re-identification module**: returns positions of co-identical instances of the query in subsequent frames in a query image
  - **Identity re-identification**: given an image of a query identity, a re-identification (re-id) algorithm ranks every image in a gallery based on its feature distance to the query identity
    - The lower the distance the higher the similarity
    - Trained neural network

Issues

- Network/compute-intensive
- Live video analytics not optimized based on cross-camera relationships
- Accuracy

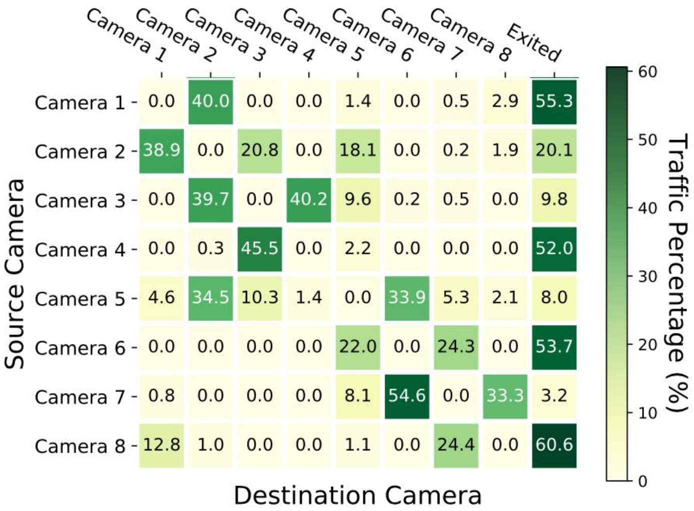
Spatio-Temporal Correlations

- **Spatial Correlations**
  - Geographical association between cameras
  - Probability that objects seen in a source camera will move next to a particular destination camera's field of view
- **Temporal Correlations**
  - Association between cameras over time
  - Probability that objects seen in a source camera will move next to a destination camera's view at a particular time

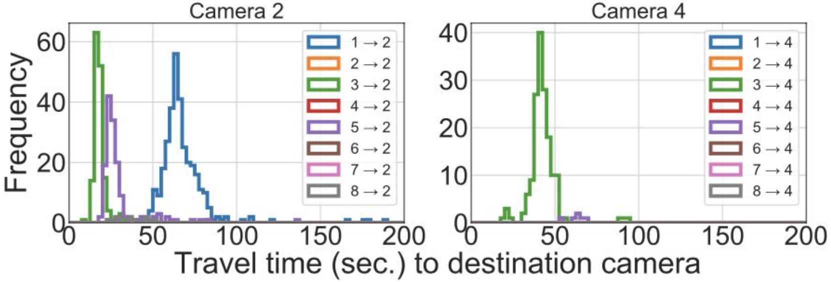
Spatio-Temporal Correlations: Duke MTMC Dataset

Figure 2. **DukeMTMC camera network [56]. Marked regions show the visual field of view of each camera.**

Spatial Correlations



Temporal Correlations



Spatula

- Spatio-Temporal Model
  - Describes the spatial and temporal correlation between cameras
- Forward Analysis
  - Real-time inference on live videos and history video
- Replay Analysis
  - Search over some history videos for error correction
- Costs proportional to the number of cameras that the queried object appears in at any point in time, and not the total number of deployed cameras

Spatula Architecture

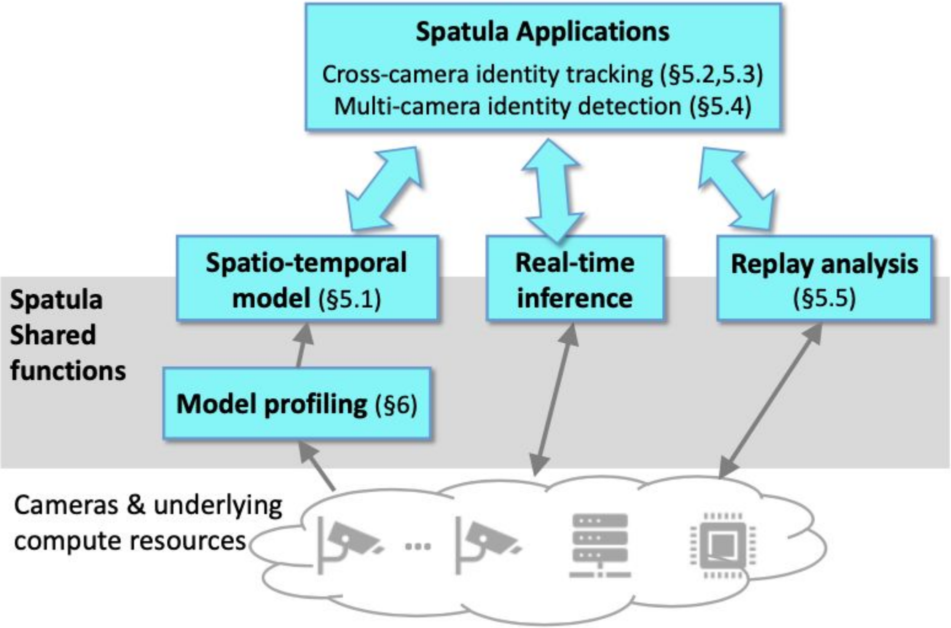


Figure 5. **Architecture of Spatula.**

Spatial Correlations

The degree of spatial correlation S between two cameras c_s, c_d is quantified by the ratio of: (a) the number of individuals leaving the source camera's stream for the destination camera, $n(c_s, c_d)$, to (b) the total number of entities leaving the source camera:

$$S(c_s, c_d) = \frac{n(c_s, c_d)}{\sum_i n(c_s, c_i)}$$

Spatula exploits spatial correlations by prioritizing cameras that are highly correlated to the last camera where the queried identity was spot.

Temporal Correlations

The degree of temporal correlation T between two cameras c_s, c_d during a window $[t_1, t_2]$ is the ratio of: (a) individuals reaching c_d from c_s within a duration window $[t_1, t_2]$ to (b) total individuals reaching c_d from c_s:

$$T(c_s, c_d, [t_1, t_2]) = \frac{n(c_s, c_d, [t_1, t_2])}{n(c_s, c_d)}$$

Spatula exploits temporal correlations by prioritizing the time window $[t_1, t_2]$ in which a destination camera is most correlated with the query camera.

Replay Analysis

- Go back to camera that last camera that the queried identity was seen and find all correlated cameras and time windows of correlation with thresholds decreased
- If still can't find it, search the entire camera network until the exit threshold
- To avoid delay
  - Skip frame mode
  - Parallelism mode

Algorithm

Algorithm 1 Tracking with the spatio-temporal model

input: video feeds $\{V_c\}$ for camera c,
    sp_corr$(c_s, c_d) \rightarrow \{\texttt{true}, \texttt{false}\}$
    tp_corr$(c_s, c_d, f) \rightarrow \{\texttt{true}, \texttt{false}\}$
for query $(q, f_q, c_q) \in Q$ **do**
    $q_{\text{feat}} = \text{features}(q)$ ▷ extract image features
    $f_{\text{curr}} = f_q + 1$ ▷ init current frame index
    $M_q = []$ ▷ init query match array
    phase $= 1$ ▷ start phase one
    while $(f_{\text{curr}} - f_q) \leq$ exit_t **do**
        $V_{\text{corr}} =$ **filter**(sp_corr, tp_corr, c_q, f_{curr}, V)
        frames = get_frames(V_{corr}, f_{curr})
        gallery = extract_entities(frames)
        ranked = rank_reid(q_{feat}, gallery)
        if ranked[0][dist] < match_thresh **then**
            M_q = append(M_q, ranked[0][img])
            q_{feat} = update_rep(q_{feat}, ranked[0][feat])
            $f_q = f_{\text{curr}}$
            phase $= 1$ ▷ reset to phase one
            break
        $f_{\text{curr}} = \text{increment}(f_{\text{curr}})$
        if phase $= 1$ **and** $T(c_s, c_d, [f_0, f_{\text{curr}}]) > 1 - t_{\text{thresh}}$ **then**
            $f_{\text{curr}} = f_q + 1$ ▷ reset frame index
            sp_corr = relax(sp_corr)
            tp_corr = relax(tp_corr)
            phase $= 2$ ▷ start phase two
output: matched detections $\{M_q\}$

Spatio-temporal model marked in blue

Update query representation

Initiate Replay Search

AnonCampus Testbed

- AWS DeepLens cameras
- Testbed includes five cameras connected to each other via Wi-Fi and deployed on AnonCampus (school building)
- Video analytics modules run on DeepLens's on-chip GPU and CPU
- Spatula controller is responsible for profiling and maintaining the spatio-temporal model of correlations among cameras
  - Trigger message: triggers the camera to start or stop searching for a specified query identity in its video within a specified time interval
  - Feedback message: notifies the controller on an interesting incident in real-time

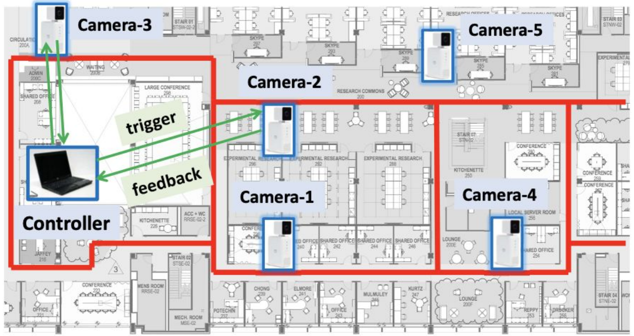


Figure 7. **Spatula testbed at AnonCampus with five AWS DeepLens smart cameras. The red lines show walkways in the building, and we learn the spatio-temporal correlation of people traversing the walkways. The *controller* and all the cameras exchange "trigger" and "feedback" messages.**

Evaluation Methodology - Datasets

- AnonCampus dataset is captured by 5 DeepLens cameras deployed in a school building
- DukeMTMC dataset is a video surveillance dataset from eight cameras in the Duke University
- Porto dataset is a simulated dataset generated from GPS trajectories obtained from 442 taxis running in the city of Porto, Portugal
    - Manually pin 130 cameras at intersections of the city
- Beijing dataset is a simulated large dataset from 17,621 GPS trajectories
    - Manually pin 600 cameras at intersections of the city

Evaluation Methodology - Models & Workload

- Models
  - Apply the MTMC tracker to label a subset of the dataset
  - Implement algorithm
  - Use a ResNet-50-based implementation of person re-id, trained in PyTorch at inference time
- Workload
  - Run a set of tracking queries drawn from each of the test query partition datasets
  - Each tracking query consists of multiple iterations

Evaluation Methodology - Metrics

- Compute cost – number of video frames processed, aggregated over all queries
- Network cost – average network bandwidth usage of transmitting encoded videos required by search algorithms
- Recall – ratio of query instances retrieved to all query instances in dataset
- Precision – ratio of query instances retrieved to all retrieved instances
- Delay (sec.) – lag between position of tracker and current video frame, in seconds, at the end of a tracking query

Evaluation Methodology - Baselines

1. Baseline (all) - searches for query identity in all cameras at every frame step (no spatio-temporal filtering)
2. Baseline (GP) - searches for query identity only in the cameras that are in geographical proximity to the query camera at every frame step
3. Spatula - searches for query identity only on cameras that are currently spatio-temporally correlated with camera with query detected

Results

- Spatula significantly outperforms both baselines by
    - Reducing compute and network cost
    - Improving precision, while maintaining comparable recall
    - Delay increase

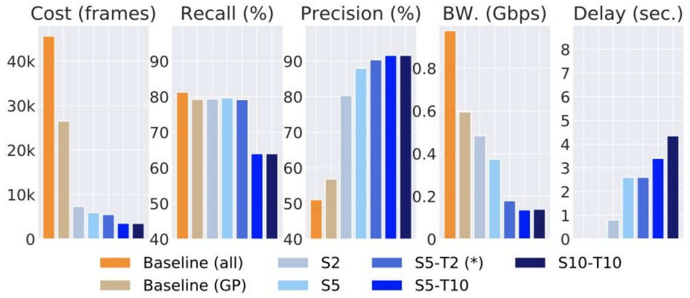


Figure 8. **Results for all-camera baseline (orange), geo-proximity baseline (tan) vs. five versions of Spatula (blues) on the DukeMTMC dataset. We argue S5-T2 (*) offers the best trade-off on all metrics.**

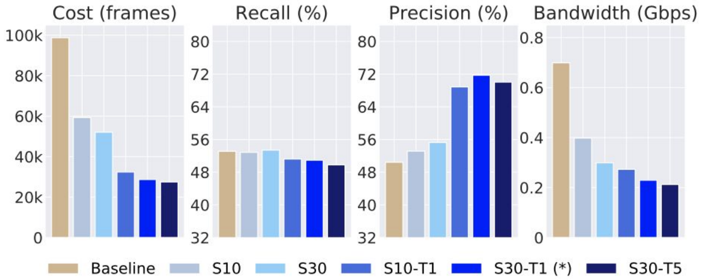


Figure 9. **Results for all-camera baseline (tan) vs. five versions of Spatula (blues) on the AnonCampus dataset. We argue S30-T1 (*) offers the best trade-off on all metrics.**

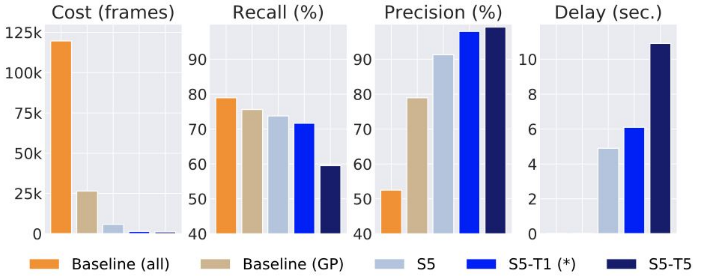


Figure 10. **Results for all-camera baseline (orange), geo-proximity baseline (tan) vs. three versions of Spatula (blues) on the Beijing dataset. We argue S5-T1 (*) offers the best trade-off on all metrics.**

Spatula Evaluation Highlights

Dataset	Comp. sav.	Netw. sav.	Prec.	Recall
AnonCampus	3.4x	3.0x	21.3% ↑	2.2% ↓
DukeMTMC	8.3x	5.5x	39.3% ↑	1.6% ↓
Porto	22.7x	n/a	36.2% ↑	6.5% ↓
Beijing	85.5x	n/a	45.5% ↑	7.3% ↓

Spatula at Scale

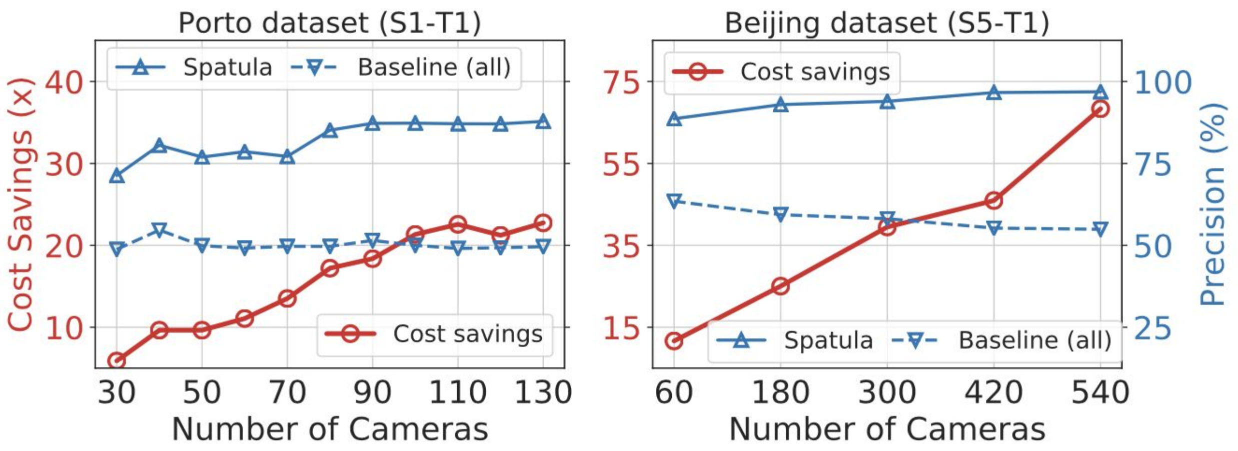


Figure 13. **Cost savings vs. number of cameras.**

Cost savings steadily grows with increasing number of cameras, achieving up to 68× lower cost than baseline (all) in Spatula S5-T1 for 540 cameras.

Results - Replay Search

- Skip frame mode - 0.5x frame sampling rate to increase throughput on historical frames (2x skip)
- Parallelism mode - 2x frame processing rate to increase throughput (increased resource usage) (2x ff)

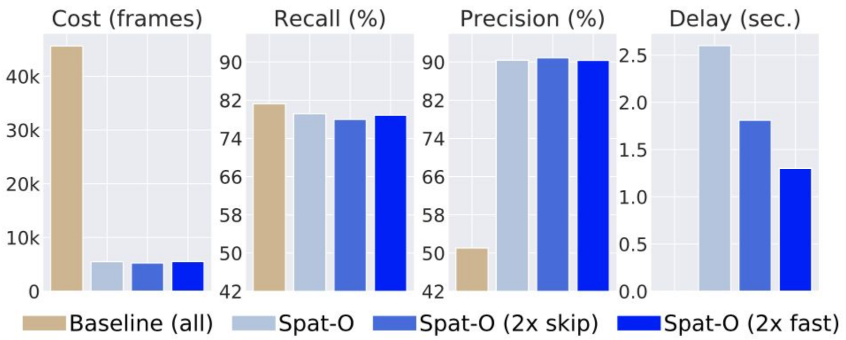


Figure 14. **Replay search. Schemes compared: baseline, Spatula-O (normal replay search), Spatula-O (2× skip), Spatula-O (2× fast-forward). Scheme 2× skip outperforms 2× fast-forward on both compute cost and delay.**

Positives

- Can be utilized for large camera deployments
- Decrease costs
- Improve precision
- Able to successfully recover from misses

Negatives

- Likelihood thresholds, that makes it vulnerable to missing “outliers”
- Model needs to be tailored to each environment
- Slight decrease in recall

Discussion

- What other types of applications could benefit from this model?
- How does a simulated data compare to a real-world dataset of the same scale?
- What kinds of issues would impact the efficacy of the model?