Spatial Keyframe Extraction of Mobile Videos for Efficient Object Detection at the Edge

George Constantinou, *Cyrus Shahabi, Seon Ho Kim*

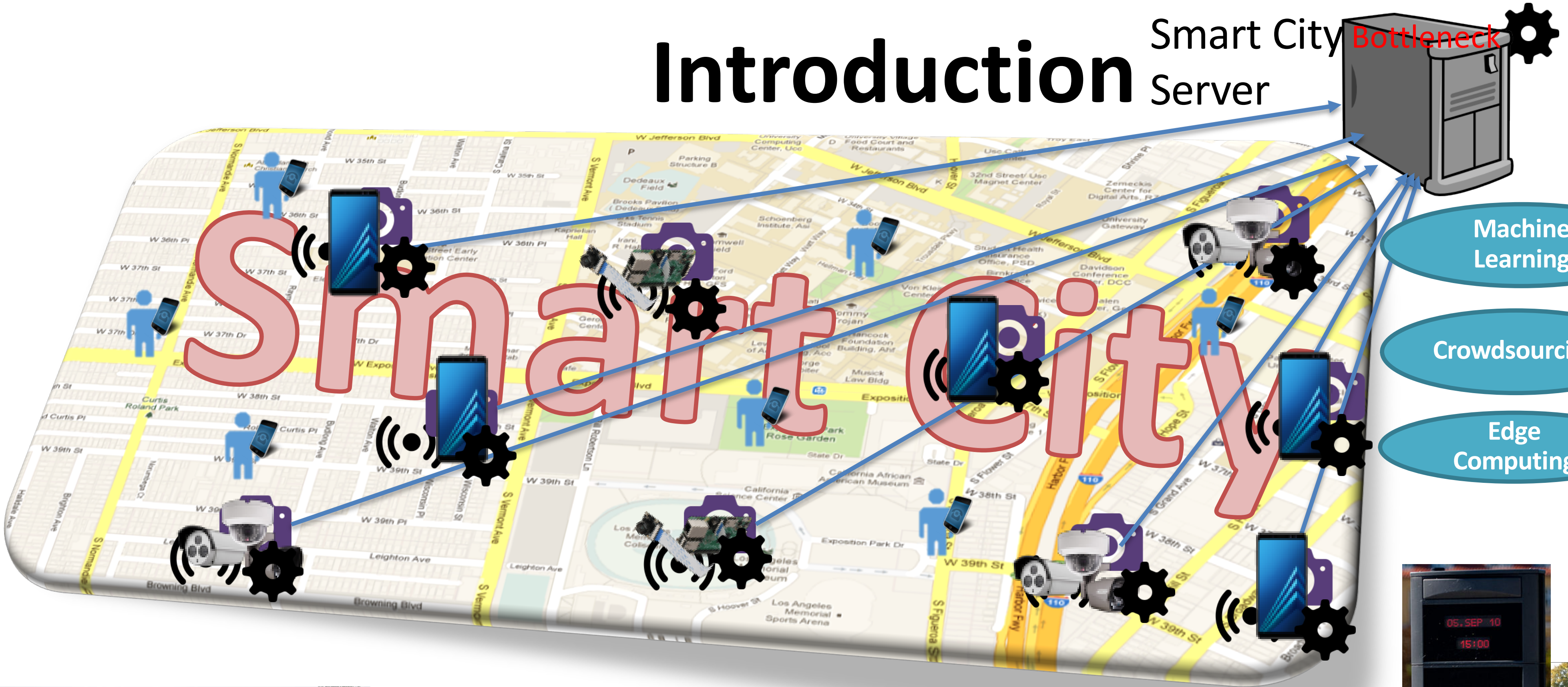
Integrated Media Systems Center (IMSC)
University of Southern California,
Los Angeles, CA, USA

Session Title: ARS-04: Image & Video Interpretation and Understanding

Outline

- ❑Introduction
- ❑Problem definition
- ❑Spatial Keyframe Extraction
- ❑Experiments
- ❑Future Directions

Introduction



- **Machine Learning** — Convolutional Neural Networks
- **Crowdsourcing** — Ubiquity of mobile users
- **Edge Computing** — Improvements in hardware and software

Platform	CPU	GPU	Inference Time (s)
Raspberry PI	0.6	No	360
PC	1.596	No	29
Low-end Server	1.386	Yes	0.2

https://www.smartcitiesworld.net/news/news/ai-equals-a-greener-life-for-trondheim-2407

Introduction

- Cameras are now able to capture videos at a frame rate of **24-60 FPS**

- A CNN can process such visual data volume **at this rate** with powerful GPU/CPU

➢ Centralized Server cannot scale as number of devices increases

Example

- Consider a smart city application for **Los Angeles Sanitation** (LASAN) Department

- LASAN installs **4-8 cameras at different angles** on a sanitary trucks to monitor and prioritize the street cleaning

- LASAN operates more than **750 trucks**

➢ The network and server will be overwhelmed

Problem definition

- Edge devices cannot feed the entire video frames to the **Convolutional Neural Network**
  - Lots of computing power
  - Lots of redundancy
  - Run one inference per few milliseconds to few minutes
- **Goal**: Select subset of video frames to feed to classification/object detection models

➢ Need a method to select “**meaningful**” keyframes

Spatial Keyframe Extraction Algorithm

- Properties
  - Leverages the **geospatial metadata** of video frames
    - Becomes a coverage problem
  - Considers the **residual overlap weight** of selected frames
  - Supports **processing capacity** of heterogeneous devices

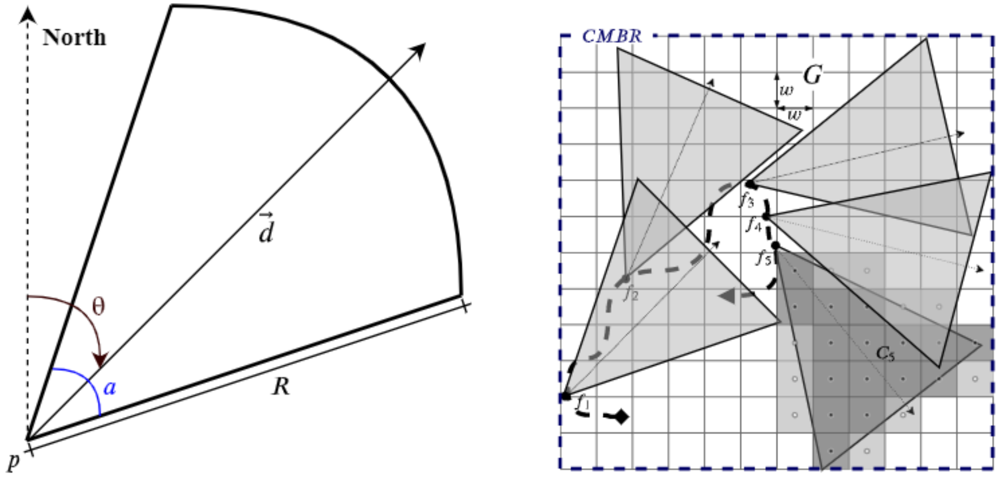
➢ But how to select frames without ”seeing” the frames?

Algorithm 1 Greedy-SKE

procedure GREEDY($\mathcal{F}, \mathcal{C}, W, f, B$)
 $\mathcal{S} = \emptyset$ ▷ Solution set
 $U = \mathcal{C}$ ▷ Cell Universe
 $\mathcal{F}' = \mathcal{F}$ ▷ Candidate FOVs
 while $|\mathcal{S}| \leq B$ **do** ▷ Up to B FOVs
  $bestFOV = bestCellSet = null, bestWeight = 0$
  for all $f_i \in \mathcal{F}'$ **do**
   $FOVCells = getFOVCells(f)$ ▷ Cell Set $\mathcal{C}_i$
   $uncoveredCells = FOVCells \cap U$
   $weight = computeWeight(f, uncoveredCells)$
   $coveredCells = FOVCells \setminus U$
   if $coveredCells \neq \emptyset$ **then**
    $weight = computeResidual(f, \mathcal{S}, coveredCells)$
   end if
   if $weight > bestWeight$ **then**
    $bestFOV = f$
    $bestCellSet = FOVCells$
    $bestWeight = weight$
   end if
  end for
  if $bestFOV == null$ **then break**
  end if
  $\mathcal{S} = \mathcal{S} \cup bestFOV$
  $\mathcal{F}' = \mathcal{F}' \setminus bestFOV$
  $U = U \setminus bestCellSet$
 end while
 return $\mathcal{S}$ ▷ The greedy solution to MWOCP
end procedure

Definitions – FOV & CMBR

- Sensor-equipped cameras can enrich the captured video:
  - with GPS location (up to per **second**)
  - with Camera viewing direction (per few **microsecond**)



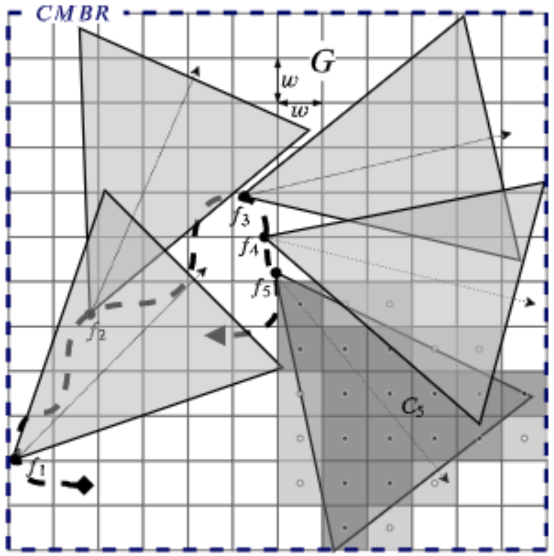
(a) The 2-D FOV model. (b) 5 FOVs and their $CMBR$.

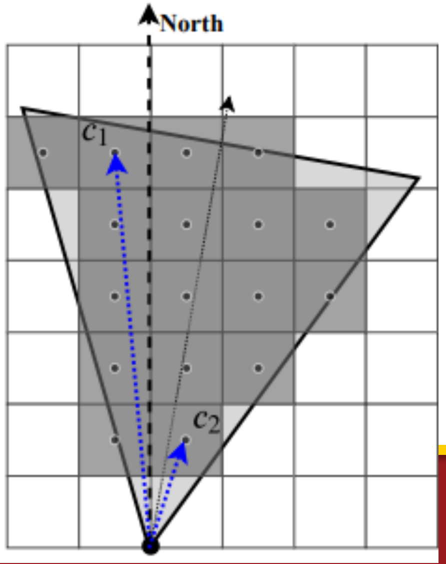
- **Field-Of-View**: A video v is represented as a set of individual video frames (or FOVs) $F = \{f_1, f_2, \dots, f_i, \dots, f_n\}, \boldsymbol{f_i} =< \boldsymbol{p}, \boldsymbol{\theta}, \mathbf{R}, \boldsymbol{\alpha} >$ ordered by the time t_i at which the frame was captured.
- **Coverage Minimum Bounding Rectangle**: Given a set of FOVs F, the CMBR is the minimum bounding box which contains all FOVs.

Definitions – Grid & Cells

- **Coverage Grid**: Given a CMBR and cell size w, we partition the CMBR into a set of square cells $G = \{c_1, c_2, \dots, c_m\}$ of width w forming the Coverage Grid.
- **Cell Set**: Given a set of FOVs F and the grid G, the Coverage Cell Set $C \subseteq G$ contains all the cells which are covered by at least one FOV.
- **Cell Spatial Weight**: The between the camera location of FOV f_i and the cell center c_j

$$w_{i,j} = \begin{cases} 1 - \dfrac{d(f_i.p, c_j.p)}{f_i.R}, if\ d(f_i.p, c_j.p) \le f_i.R \\ 0, otherwise \end{cases}$$





Definitions – Overlap function & residual weight

- **Cell Overlap Weight Function**: A function $f: X \rightarrow Y$ which defines what is the new spatial weight of the cell when multiple FOVs cover it and are selected by the solution.
  - $X = \left\{x \in \mathbb{R}^{\left|F_j'\right|} | x = w_{i,j}, f_i \in F_j', c_j \in C_i\right\}$
  - $Y = \{y \in \mathbb{R} |\ 0 \ \leq y \leq \ 1\}$

- **Residual Overlap Weight**: For a current frame selection S, the residual overlap weight for f_i and its cells C_i is computed as follows:
  - for cells $c_j \in C_i$ not covered by any other FOV already in S, the residual weight $w_{i,j}^r$ is equal to $w_{i,j}$
  - Otherwise, use f to calculate $w_{i,j}^o$ assuming f_i was added in S, $w_{i,j}^r = w_{i,j}^o \ - \ w_{i,j}$

➢ A new FOV increases the total weight by only the weight difference

➢ Now we can define as a coverage problem

Maximum Weighted Overlap Coverage Problem

- Given a set of FOVs $F = \{f_1, f_2, \dots, f_i, \dots, f_n\}$, the weights $w_{i,j}$, the cell overlap weight function f, the set of covered cells $C_i = \{c_1, c_2, \dots, c_m\}$ for each FOV, the maximum budget for frames B, the *Maximum Weighted Overlap Coverage Problem* (MWOCP) finds a subset F' s.t. $|F'| \le B$ which maximizes the weighted sum of covered cells in the sets F_j'.

- MWOCP is NP-Hard
  - Reduction from Maximum Coverage Problem (MCP)

➢ Greedy-SKE solution is proposed

Greedy-SKE Time Complexity

- n = |F| FOVs
- m = |C| covered grid cells
- B budget
- Find uncovered cells O(m’), m’ << m
- Find covered cells O(m’’), m’’ << m
- Compute residual O(m’’)
- Runs in $O(B \cdot n \cdot \max(\text{m'}, \text{m''})) \leq O(n^2 \cdot m)$

Algorithm 1 Greedy-SKE

procedure GREEDY($\mathcal{F}, \mathcal{C}, W, f, B$)
  $\mathcal{S} = \emptyset$ ▷ Solution set
  $U = \mathcal{C}$ ▷ Cell Universe
  $\mathcal{F}' = \mathcal{F}$ ▷ Candidate FOVs
  while $|\mathcal{S}| \leq B$ **do** ▷ Up to B FOVs
    $bestFOV = bestCellSet = null, bestWeight = 0$
    for all $f_i \in \mathcal{F}'$ **do**
      $FOVCells = getFOVCells(f)$ ▷ Cell Set $\mathcal{C}_i$
      $uncoveredCells = FOVCells \cap U$
      $weight = computeWeight(f, uncoveredCells)$
      $coveredCells = FOVCells \setminus U$
      if $coveredCells \neq \emptyset$ **then**
        $weight = computeResidual(f, \mathcal{S}, coveredCells)$
      end if
      if $weight > bestWeight$ **then**
        $bestFOV = f$
        $bestCellSet = FOVCells$
        $bestWeight = weight$
      end if
    end for
    if $bestFOV == null$ **then** break
    end if
    $\mathcal{S} = \mathcal{S} \cup bestFOV$
    $\mathcal{F}' = \mathcal{F}' \setminus bestFOV$
    $U = U \setminus bestCellSet$
  end while
  return $\mathcal{S}$ ▷ The greedy solution to MWOCP
end procedure

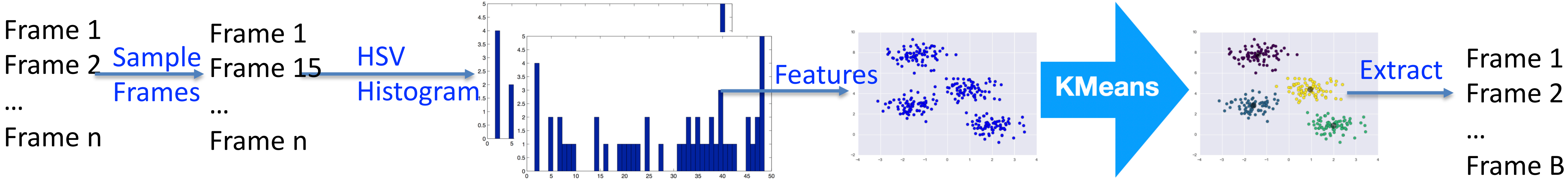
Baselines - Greedy-Naive

- Uses a max-heap to get cells in order based on their cumulative spatial weight of all FOVs. For the current cell c_j, a random FOV $f_i \in F_j$ is selected and added to the solution S.

- Additionally, all cells $c_l \in C_i$ are removed from the heap. The algorithm stops when budget B is reached or the heap is emptied.

Baselines - Clustering

- A set of frames are sampled every half-second.
- Generate histogram of 50 bins is constructed from the HSV color space [1] (20-20-10 bins for each component, respectively)
- Use histogram as feature vector
- Use k-means with k=B, the budget value
- Extract the closest frame from each cluster centroid

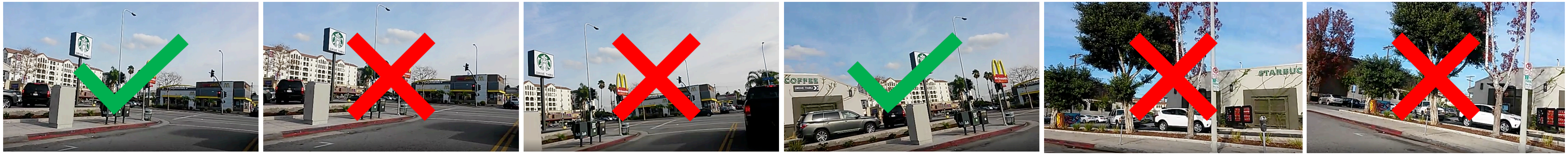


[1] Sandra Eliza Fontes de Avila, Ana Paula Brand̃ ao Lopes, Anto-nio da Luz, and Arnaldo de Albuquerque Ara ́ujo, "Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method, "Pattern Recognition Letters, vol. 32,no. 1, pp. 56 – 68, 2011, Image Processing, Computer Vision and Pattern Recognition in Latin America.

Baselines - Temporal

- Selects frames based on a predefined sampling rate t_s
  - e.g., 2 frames per second

- t_s can be adjusted in a way, such that it matches the processing capacity of an edge device



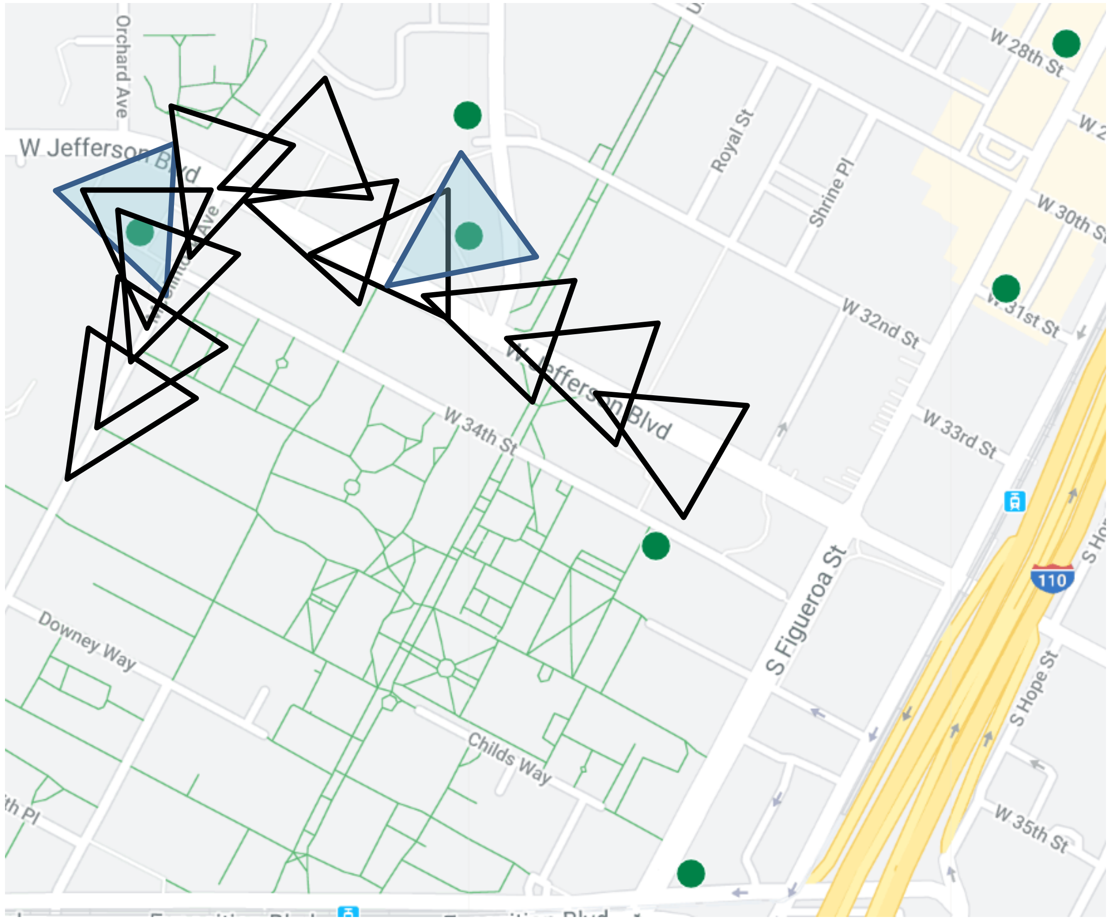
Baselines - Trajectory-SKE

- Selects frames based on the camera location of the FOV metadata.
- Sort frames by capture time
- A user-defined radius threshold t_r is used to determine whether the camera location of the frame is farther than the previous frame's radius
- Sampling the trajectory by adjusting t_r ensures that the selected frames are captured at different locations

Experimental Setup

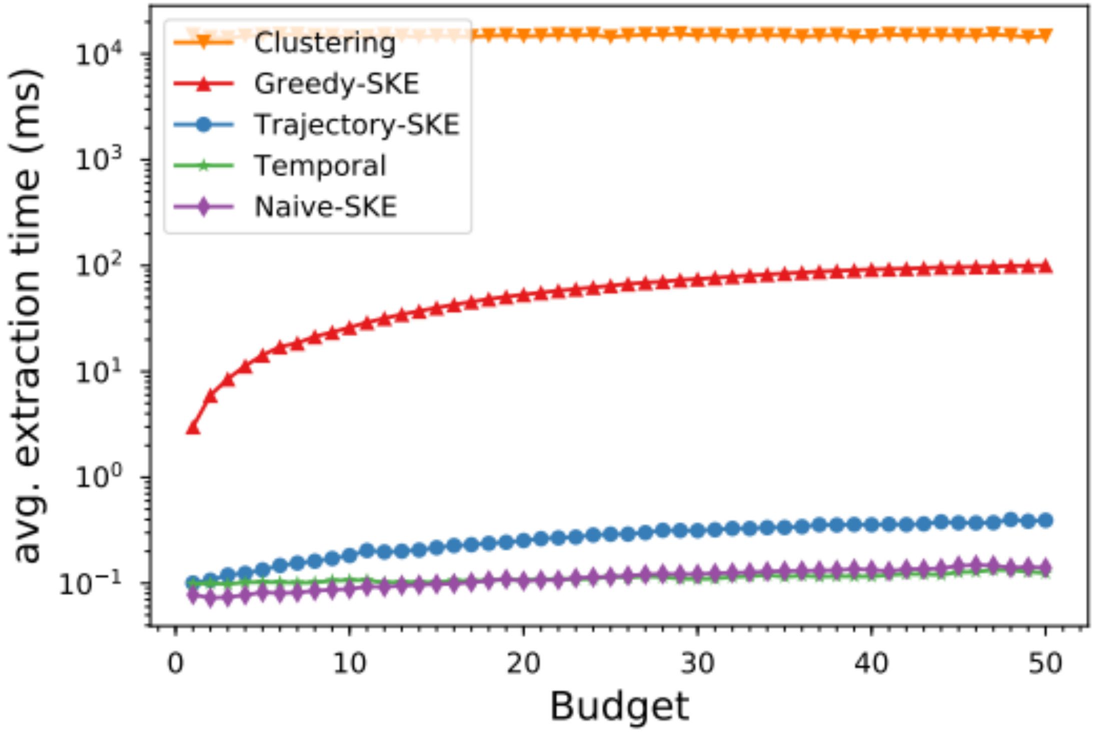
- Collected 25 FHD videos at 30FPS along with their FOV metadata
  - generating 69K frames (2872 frames per video on average)
- All videos were recorded so that they intentionally contain some frames that capture a **Starbucks coffee shop**
  - Q: how efficiently detect Starbucks logos from the collected videos?
- Used **Google Vision API** to detect the Starbucks logo for each frame in every video and log the detected frames with a confidence≥70%
  - resulting to 5.5K frames (~8% of total frames)
- Experiments on **Ubuntu Desktop** and **Raspberry Pi 3 Model B**

Logo Visibility Example in Starbucks experiment

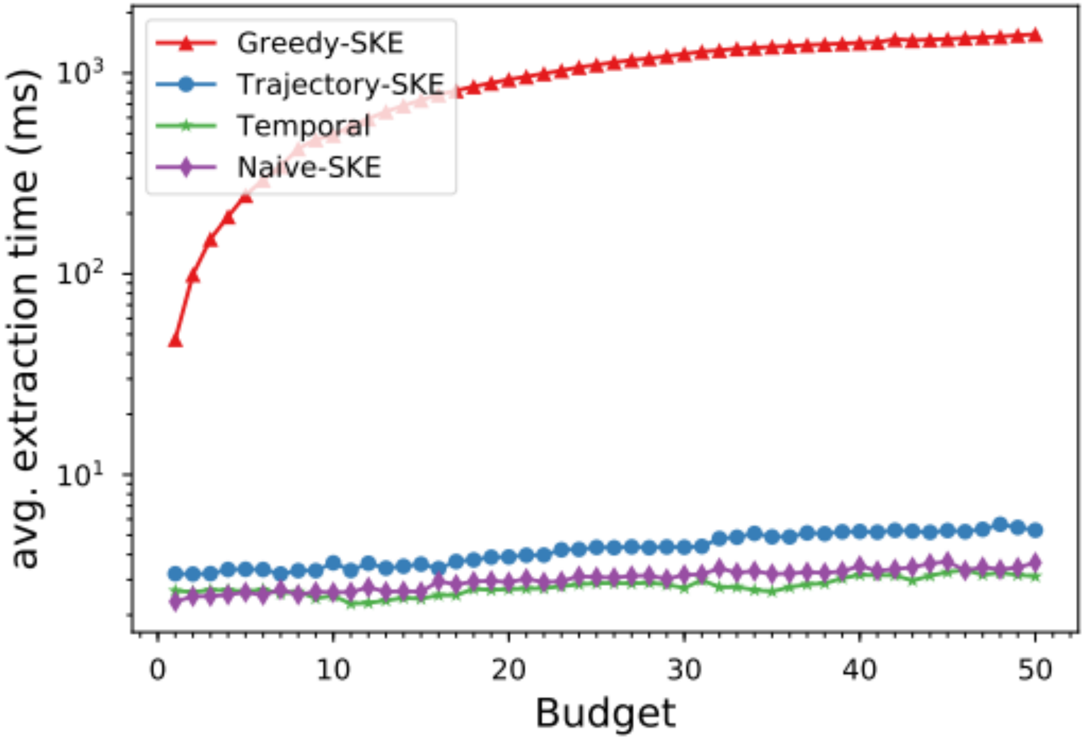


Experimental Results - Performance

Desktop



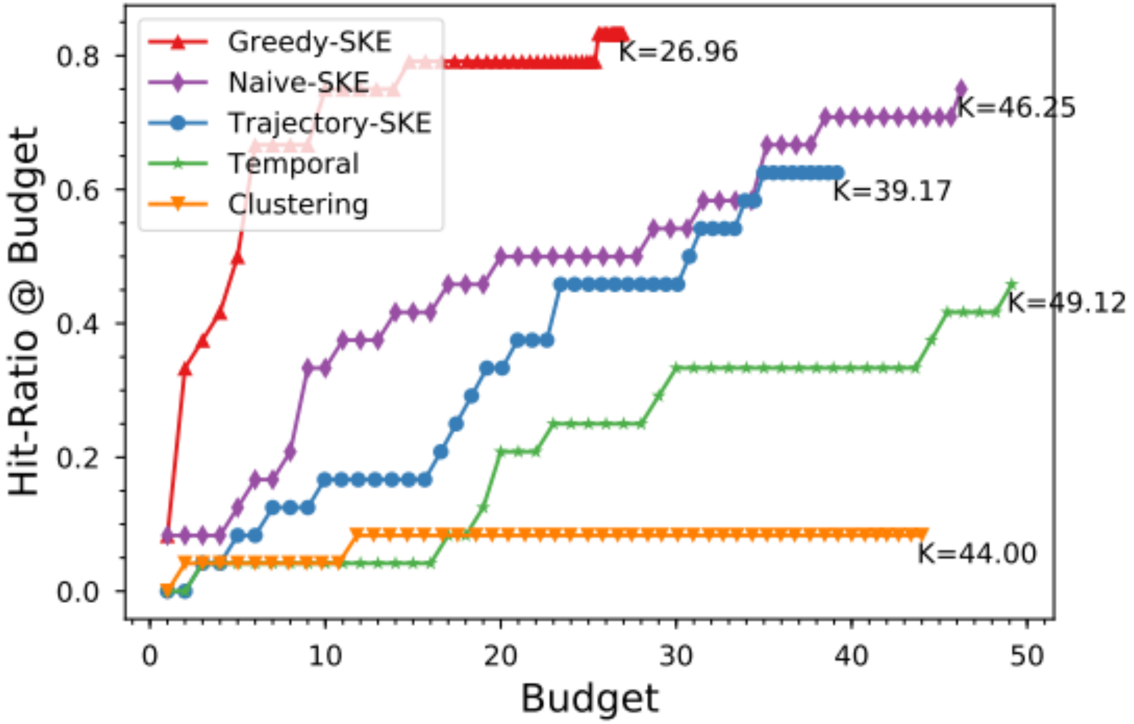
Raspberry Pi

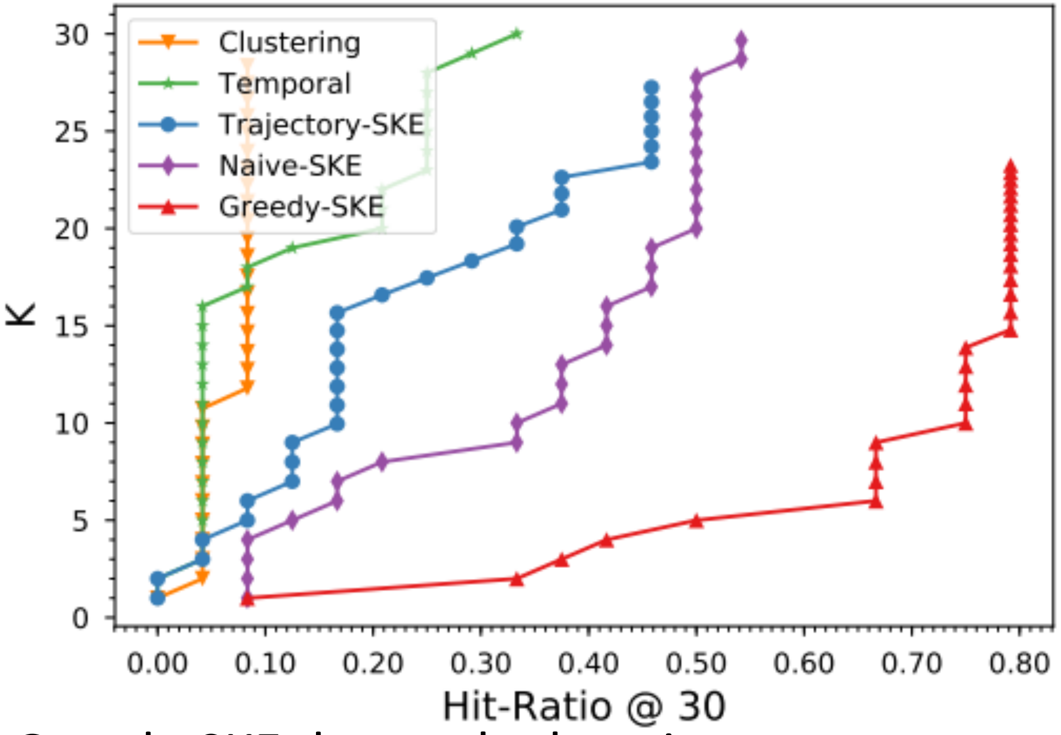


- Clustering approach suffers the most
  - 2x slower
- Greedy-SKE needs 1sec on RPI, 100ms on desktop for 50 frames

USC Viterbi
School of Engineering
Integrated Media Systems Center

Experimental Results – Spatial Weight Impact





- K < B observation
- Greedy-SKE outperforms others

- Greedy-SKE detect the logo in:
  - 66% of videos w/ 6 frames in 17ms
  - 75% of videos w/ 10 frames in 26ms
  - 80% of videos w/ 15 frames in 39ms

Conclusion

- Use spatial metadata to speedup frame selection
- Introduce **Maximum Weighted Overlap Coverage Problem**
  - Greedy solution is fast even on resource-constraint devices
- Experimental results on real video dataset show effectiveness of approach

THANK YOU!