Revealing cluster formation over huge volatile robotic data

Nikos Mitsou[1,3] , Eirini Ntoutsi[2]
Dirk Wollherr[1] , Costas Tzafestas[3]
Hans-Peter Kriegel[2]

1

2
LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

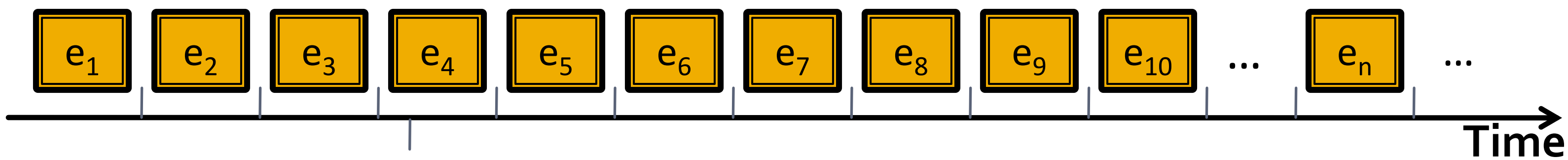
3
National
Technical
University of
Athens

Outline

- Motivation
- The robotic stream
- Stream clustering
- Experiments
- Conclusions and outlook

Data streams

- What is a data stream
  - Possible infinite sequence of elements arriving at a rapid rate



- Characteristics
  - Huge amounts of data → only a small amount can be stored in memory
  - Arrival at a rapid rate → need for fast response time
  - The generative distribution of the stream might change over time → adapt and report on changes
- Applications
  - Telecommunications, Banks, Health care systems, WWW, Robotics

Robotics

- A number of successful applications so far
  - robotic cars, e.g. Mars explorer
  - medical robotics
  - in everyday life , e.g. cleaning robots
- Due to advances in hardware and software, robotics is expanding quickly in many fields
- Key challenge: To interact with the environment, huge amounts of data are received by a robot and have to be processed

Data arrives in high rates and in huge amounts. A typical case of stream!

Outline

- Motivation
- The robotic stream
- Stream clustering
- Experiments
- Conclusions and outlook

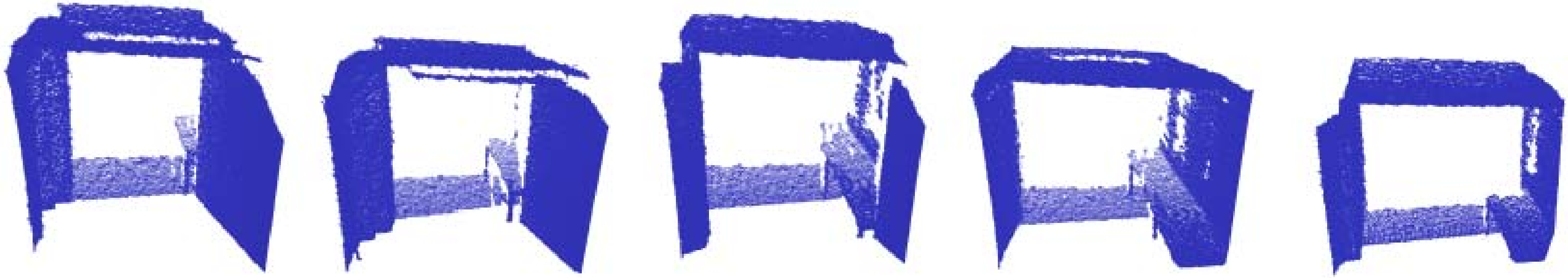
Our scenario – I

- A mobile robot
  - is moving in an unknown environment e.g. office
  - continuously scans the environment though its sensors

(a) The environment

- Each robot scan produces one *point cloud*
  - A set of 3D points – a noisy sampling of the environment
  - Kinect for Xbox 360 was used in our experiments

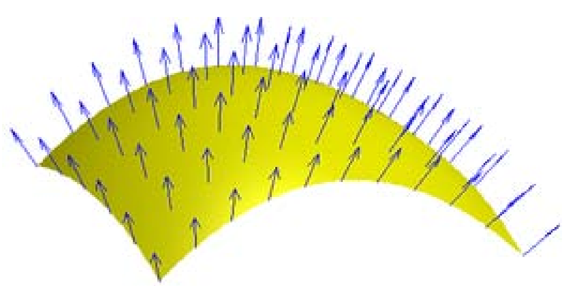
- The complete environment is “revealed” gradually

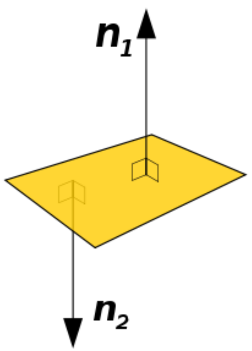


(b) Scan 1 (c) Scan 2 (d) Scan 3 (e) Scan 4 (f) Scan 5

Our scenario – II

- Point clouds contain dense spatial information but
  - do not offer any ‘direct’ information about the structure of the environment
  - do not allow for geometric interpretation and abstraction
- Surfaces / Planes
  - Are typically used as higher level representations of the environment
  - To extract a plane from 3D points, the notion of normal vectors is used
- Normal vectors
  - They describe the orientation of a plane.
  - The normal vector of a point *p* can be calculated by computing the total least square plane fitting the points $p \cup N(p)$.





The problem

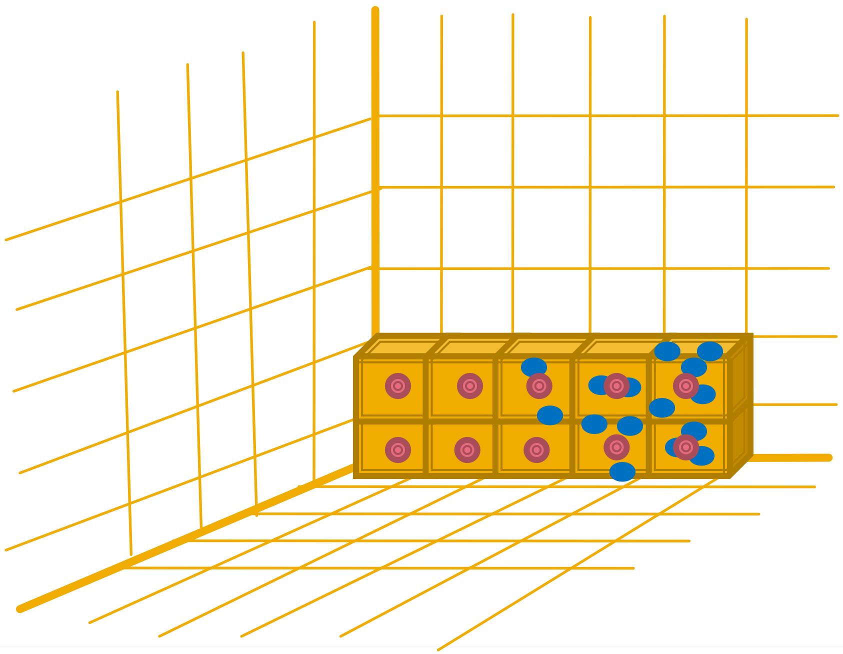
- A stream of 3D scans S_1, S_2, …, S_t,… arriving at timepoints t_1, t_2, …, t_t,…
- Each scan describes a partial view of the environment
- Goal: Online detection of object formations over time as the robot navigates in the environment.
  - Due to the partial view of the scans, an object might span in multiple consecutive scans!

Outline

- Motivation
- The robotic stream
- Stream clustering
- Experiments
- Conclusions and outlook

3D grid representation

- 200.000 to 300.000 points in every scan → for abstraction, a grid structure (set of units) is adopted
- Density of a unit d(u) = # of points inside the unit



Unit neighborhood & normal vector

- Unit neighborhood in depth d, $N_d(u)$ contains:
  - all units u' for which there exists a path $(u_1, u_2, \ldots, u_d)$, $u_1 = u$, $u_d = u''$: u_{i+1} is directly connected to u_i and $1 \leq i \leq d$.
- The notion of normal vector for a point is transfered now to units.
- Unit normal vector:
  - Is estimated by computing the total least square plane fitting $u \cup N_d(u)$.

Surface clusters

- A surface cluster is a maximal set of connected dense units belonging to the same surface.
- To find surface clusters we rely on:
  - unit vicinity in the grid
  - unit orientation similarity
- Orientation similarity evaluates whether two units belong to surfaces of similar orientation

$$sim_{orient}(\overrightarrow{u_1}, \overrightarrow{u_2}) = (\overrightarrow{u_1} \cdot \overrightarrow{u_2}) / (\|\overrightarrow{u_1}\| \|\overrightarrow{u_2}\|)$$

Our method

- Our method consists of:
  - The partial cluster extraction step for computing the local/ partial clusters appearing in a single scan S_i
    - e.g. part of the wall or table
  - The global cluster extraction step for updating the global clustering
    - e.g. merge parts of the wall into one cluster

Partial cluster extraction - I

- Convert scan points into grid units
- Compute the new dense units and their normal vectors
- Extract clusters
  - Create a new cluster *clu* from a new random dense node u_{init}
  - Expand *clu* through a directly connected unit *u* if *u'* and clu have similar normal vectors
  - Continue until *clu* cannot be further expanded
- If all new dense units are visited, finish

For every scan S_i

Partial cluster extraction - II

- The normal vector of *clu* is updated based on the newly added unit *u*

$$\overrightarrow{clu'} = \langle \frac{n * \overrightarrow{clu.x} + u.x}{n+1}, \frac{n * \overrightarrow{clu.y} + u.y}{n+1}, \frac{n * \overrightarrow{clu.zn} + u.z}{n+1} \rangle$$

n: # units in the cluster

- The output of this step is a set of partial clusters for a given scan

Global cluster extraction - I

- Given the old global clustering C_{t-1} and considering the newly discovered clusters from scan S_t, how can we extract C_t?

- To update C_{t-1}, we rely on:
  - the vicinity between the partial and the global clusters
  - their orientation similarity

- Vicinity between two clusters

$$vicinity(c_i, c_j) = |\{(u, u' : u \, is \, directly \, connected \, to \, u')\}|$$

minUnits: Cluster vicinity threshold

Global cluster extraction - II

- For every partial cluster, one of the following cases might occur:
  - Starts a new global cluster (new)
  - Continues an already existing global cluster (absorption)
  - Merges two or more global clusters (merge case)

Outline

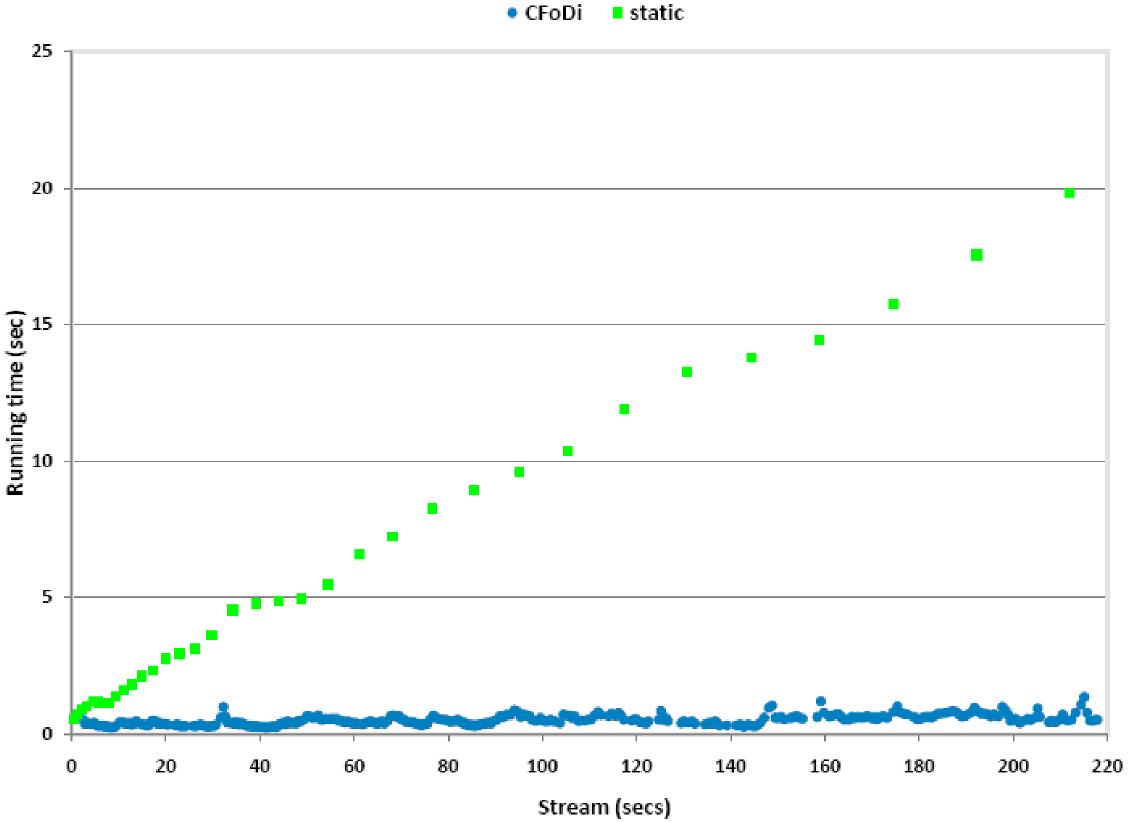
- Motivation
- The robotic stream
- Stream clustering
- Experiments
- Conclusions and outlook

Experimental settings

- The robot is navigating in an office environment
- We measure
  - execution time
  - cluster quality in comparison to the static approach
- We also visually inspect the resulting clustering
- Parameter values
  - Unit size: 8cm
  - Density threshold: 15 points
  - Orientation similarity threshold: 0.53 radians

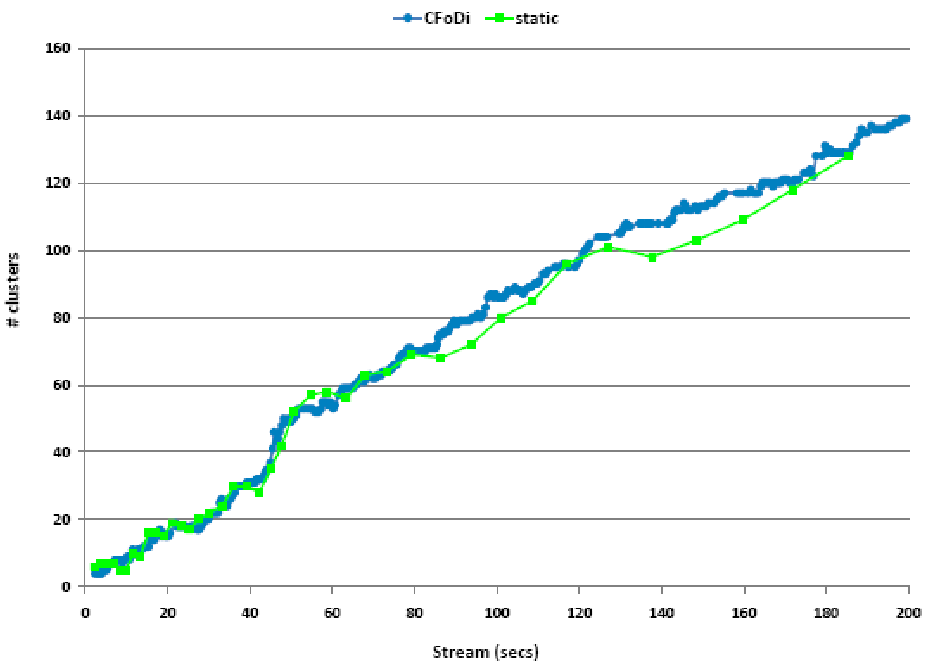
Execution time

- After 220 seconds, we processed 86,000,000 points
- Constant time for every scan 0.34 seconds

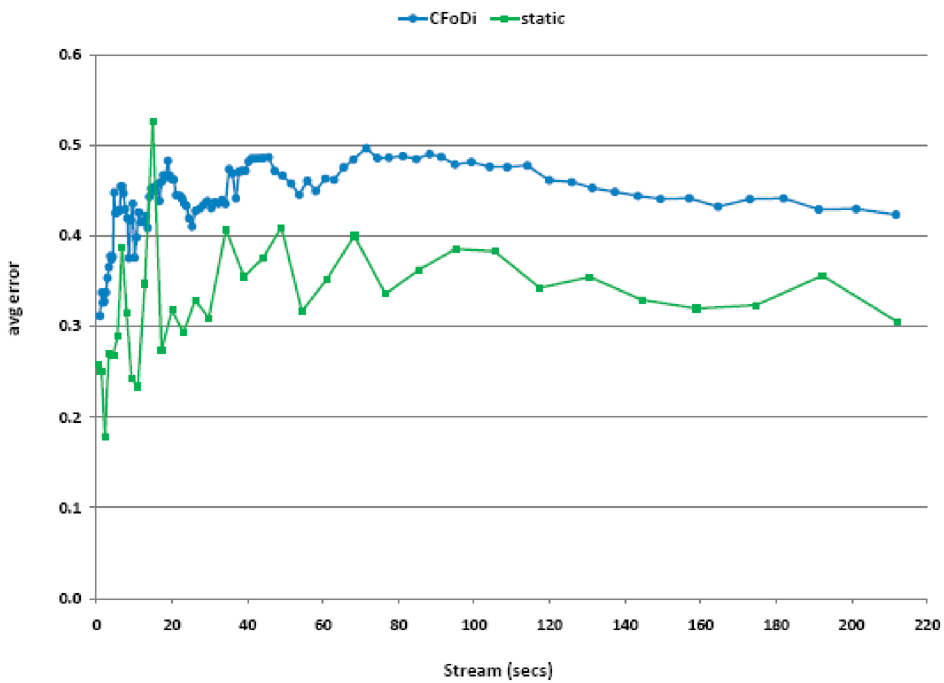


Cluster quality

- Comparison with the static approach



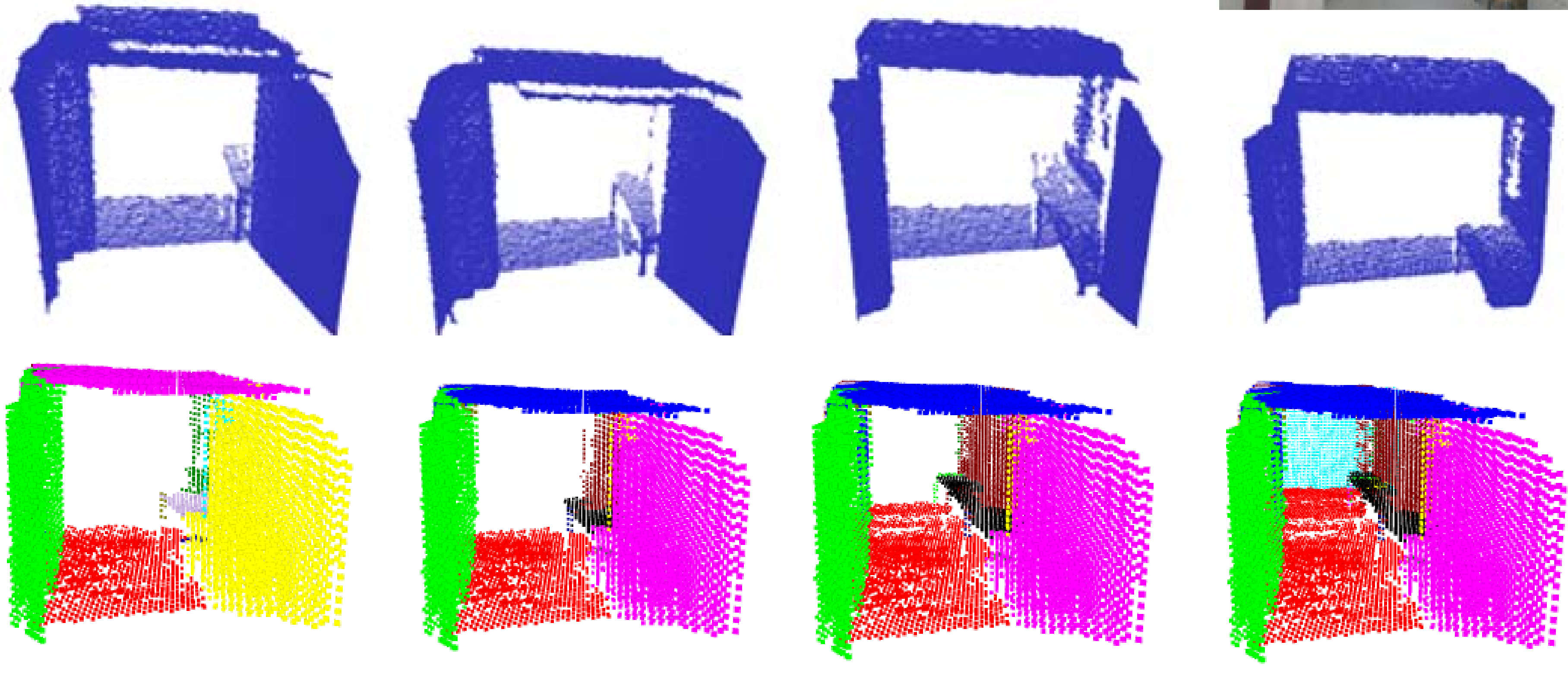
Number of clusters over time



Average normal vector error over time

Clusters visually

- Resulting clustering



(a) Scan 1 (b) Scan 2 (c) Scan 3 (d) Scan 4

Outline

- Motivation
- The robotic stream
- Stream clustering
- Experiments
- Conclusions and outlook

Conclusions & Future work

- Conclusions
  - We presented a stream clustering algorithm for detecting object formations in a robotic stream
  - The algorithm can handle high arrival rates and huge amounts of data

- Future work
  - Experiments in more complex environments
  - Dynamic selection of parameters
  - Detection of composite objects / classification

Questions?

Thank you
for your attention!

N. Mitsou is supported by
the IURO project (7th Framework Program of EU,
ICT Challenge 2 Cognitive Systems and Robotics)

http://www.iuro-project.eu

I. Ntoutsi is supported by
an Alexander von Humboldt Foundation
fellowship for postdocs

http://www.humboldt-foundation.de