# Automatic Adaptation to Sensor Replacements

**Yuan Shi**

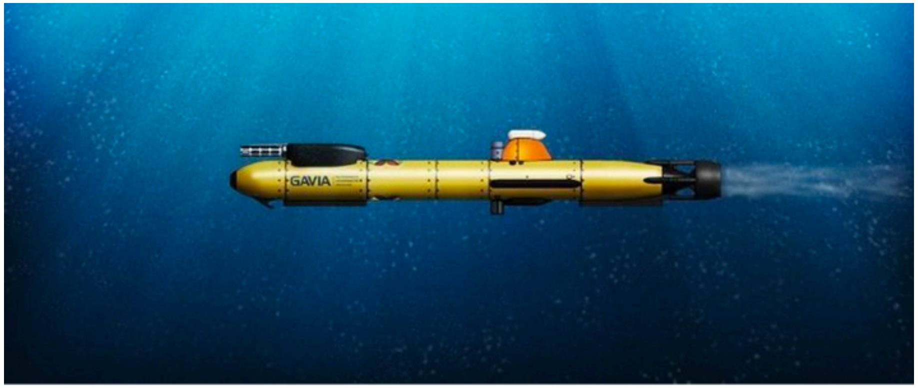
Joint work with **T. K. Satish Kumar** and **Craig A. Knoblock**

Information Sciences Institute
University of Southern California

The Florida Artificial Intelligence Research Society
FLAIRS

Supported by the United States Air Force and the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-16- C-0045

# Motivation




Unmanned Underwater Vehicle (UUV)


Self-driving Car

**Challenges for software systems**:
- Changing and uncertain environment
- System failures and changes

How to build **long-lived, survivable** software systems?
- Significantly reducing maintenance cost
- Goal of the DARPA BRASS (Building Resource Adaptive Software Systems) program

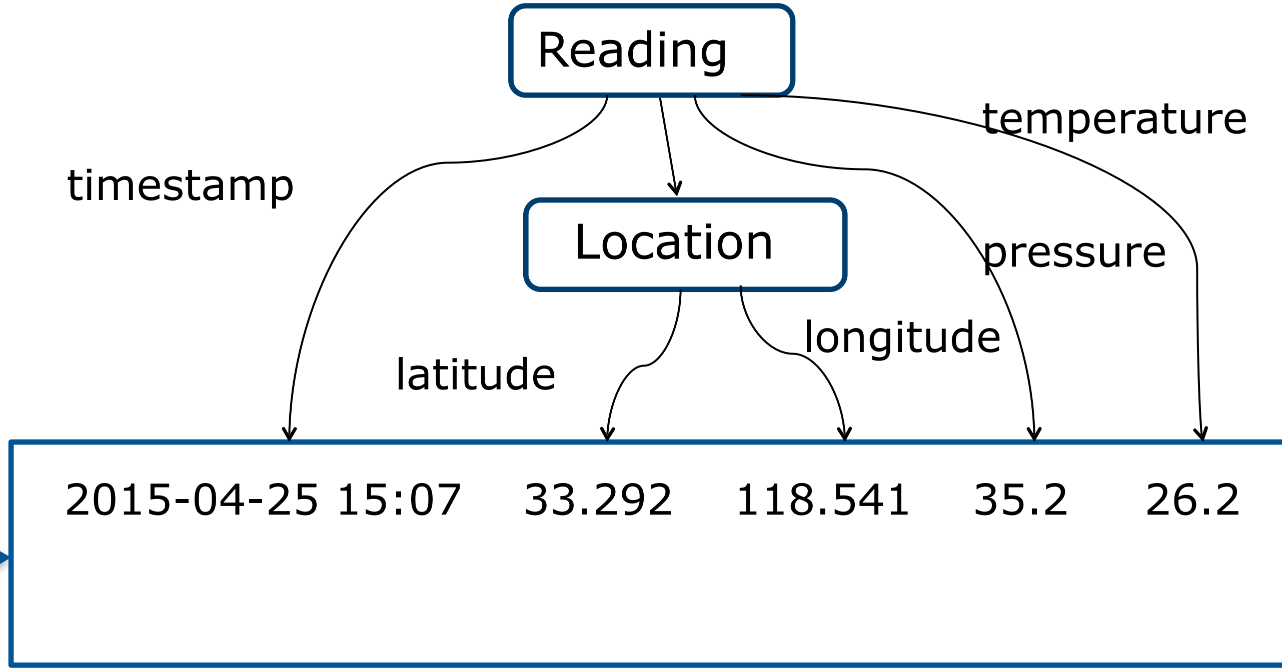
# Motivation

- Our focus: **adaptation to sensor replacements**
  - A set of sensors are replaced by new sensors
  - Causes: replacement of failed sensors, sensor upgrade, energy optimization, etc.

- Extension of our previous work [Yuan and Craig, IJCAI'17] that can be used to exploit a single new sensor

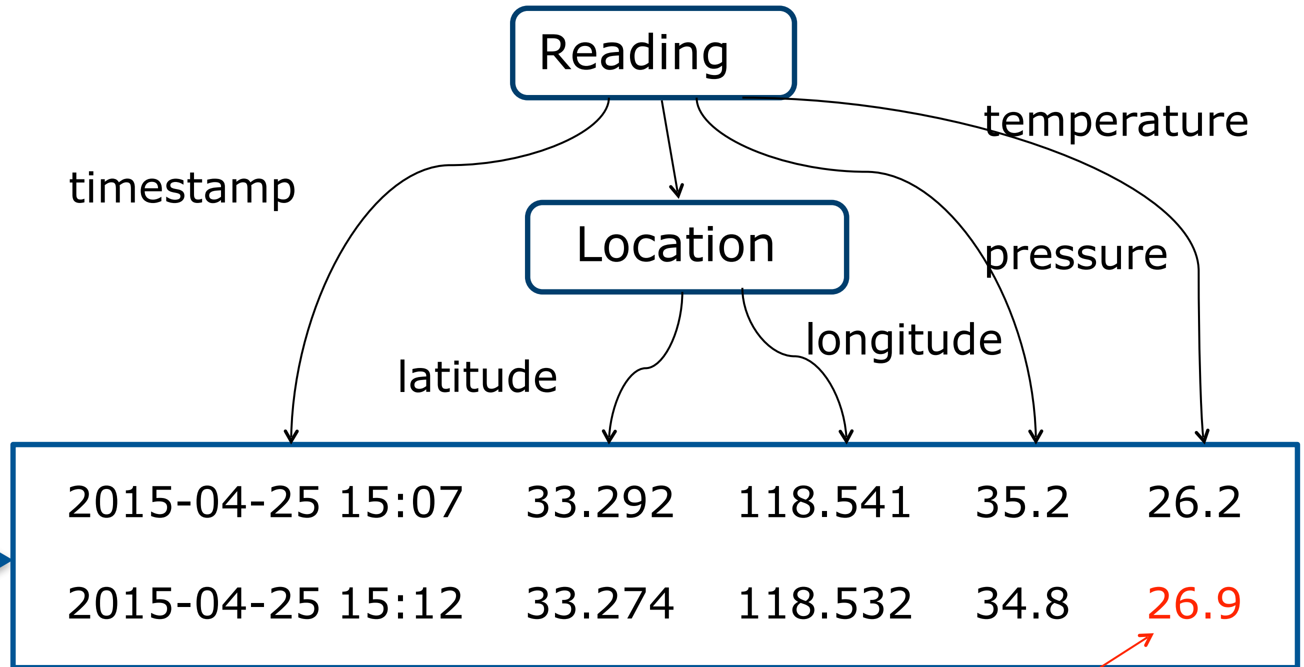
# Real-world Example of Sensor Replacement

Compound Sensor



# Real-world Example of Sensor Replacement

Compound Sensor




Replaced by a new temperature sensor

# Real-world Example of Sensor Replacement

Reading
timestamp
Location
latitude
longitude
pressure
temperature

Compound Sensor

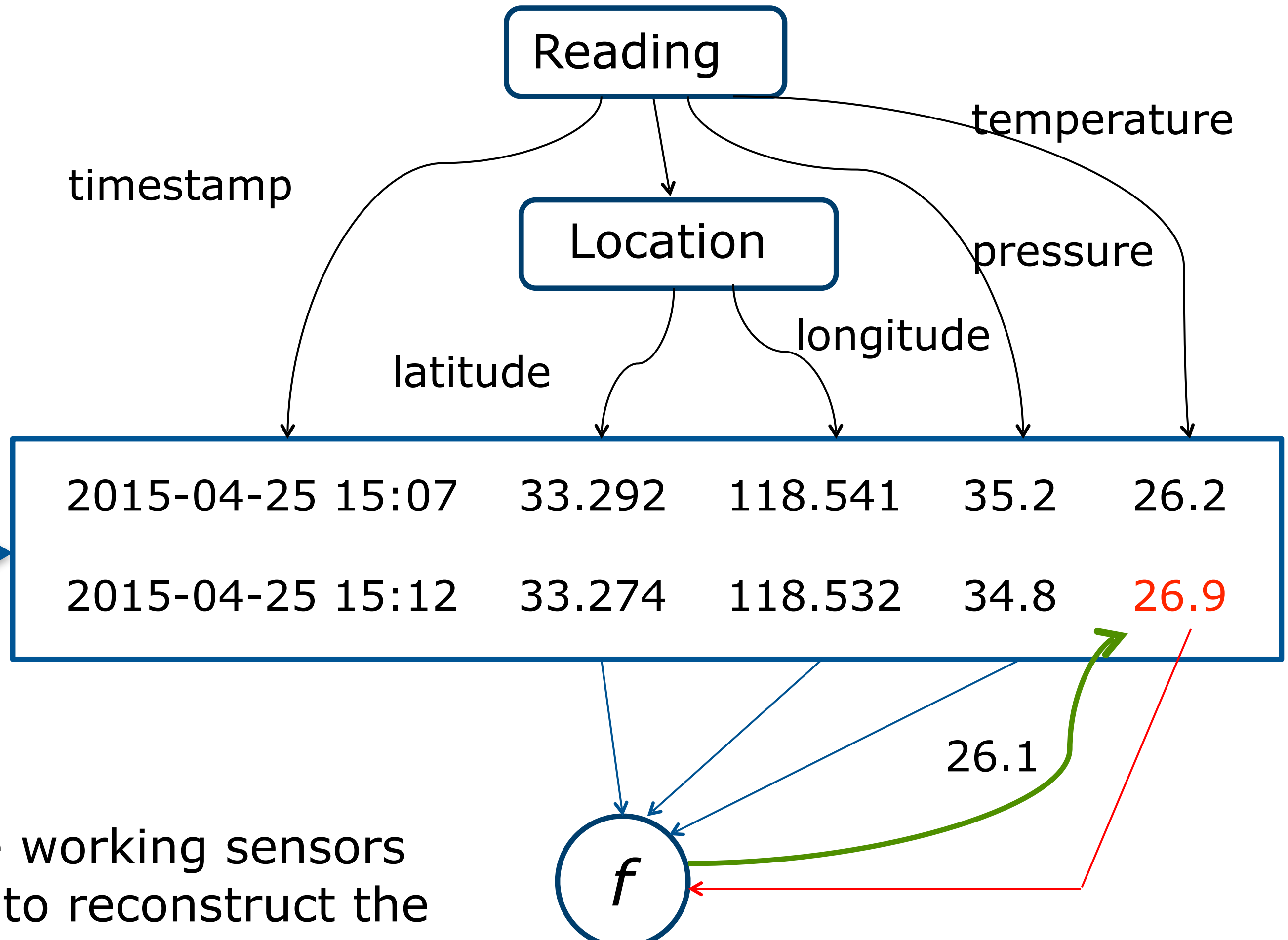
| | | | | |
|---|---|---|---|---|
| 2015-04-25 15:07 | 33.292 | 118.541 | 35.2 | 26.2 |
| 2015-04-25 15:12 | 33.274 | 118.532 | 34.8 | 26.9 |

Replaced by a new temperature sensor

Direct replacement can be bad!

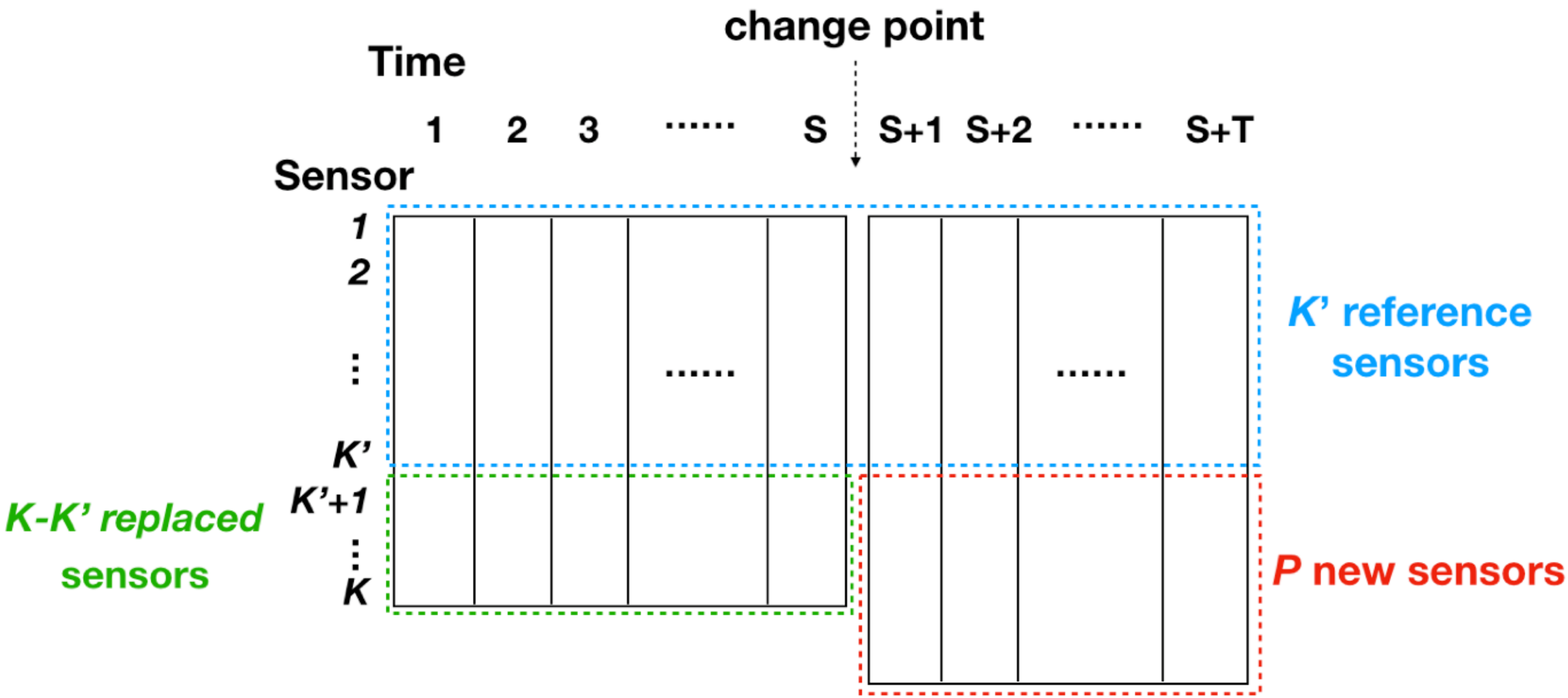
# Real-world Example of Sensor Replacement

Compound Sensor

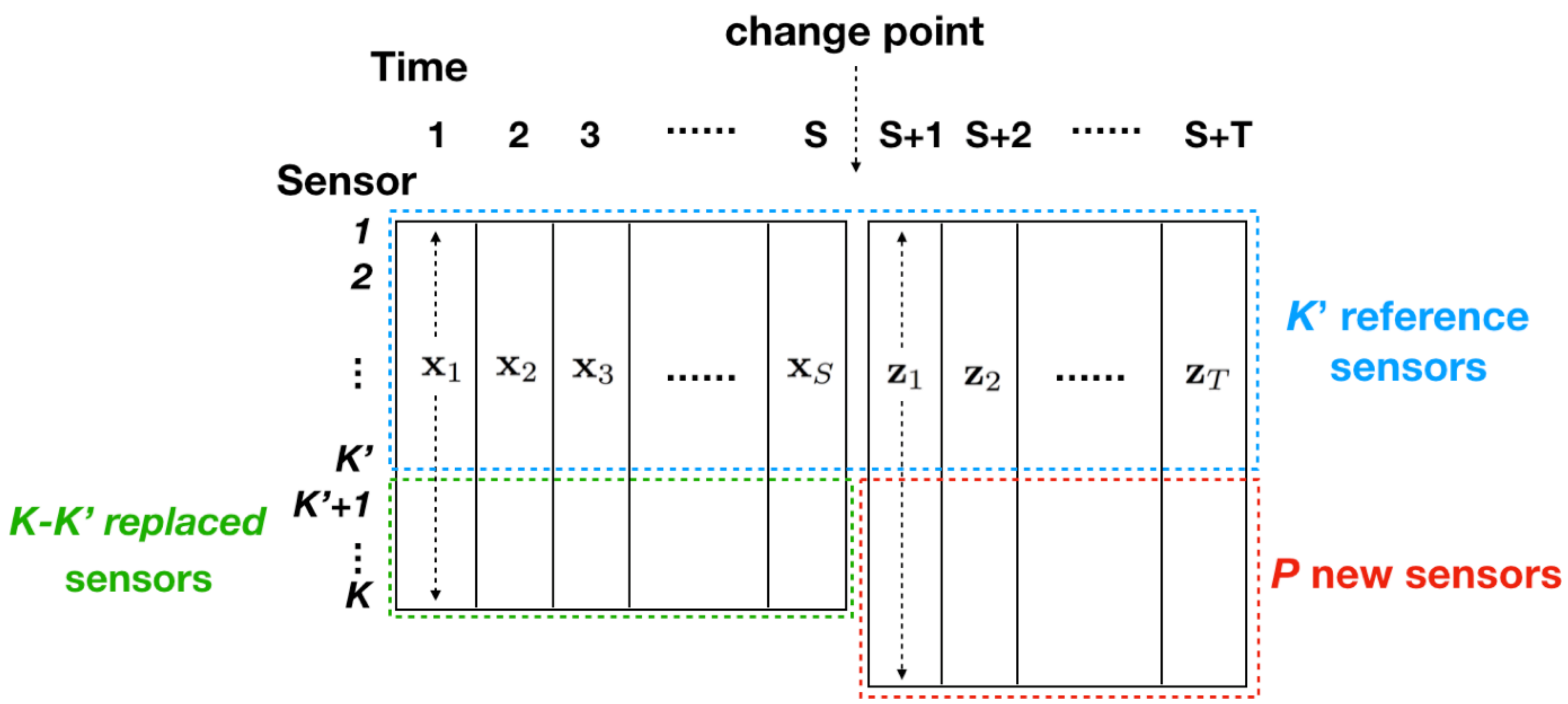



Can we combine the working sensors and the new sensor to reconstruct the original temperature?

# Notations of Sensor Replacements

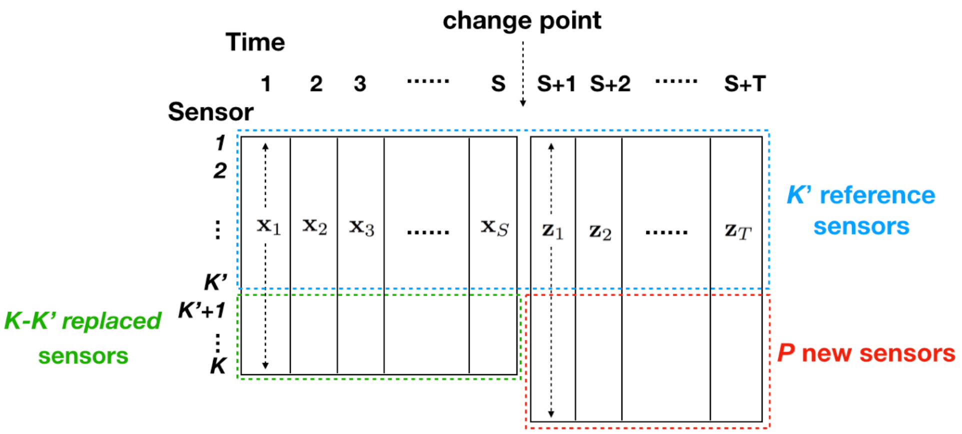


# Notations of Sensor Replacements



# Adaptation to Sensor Replacements

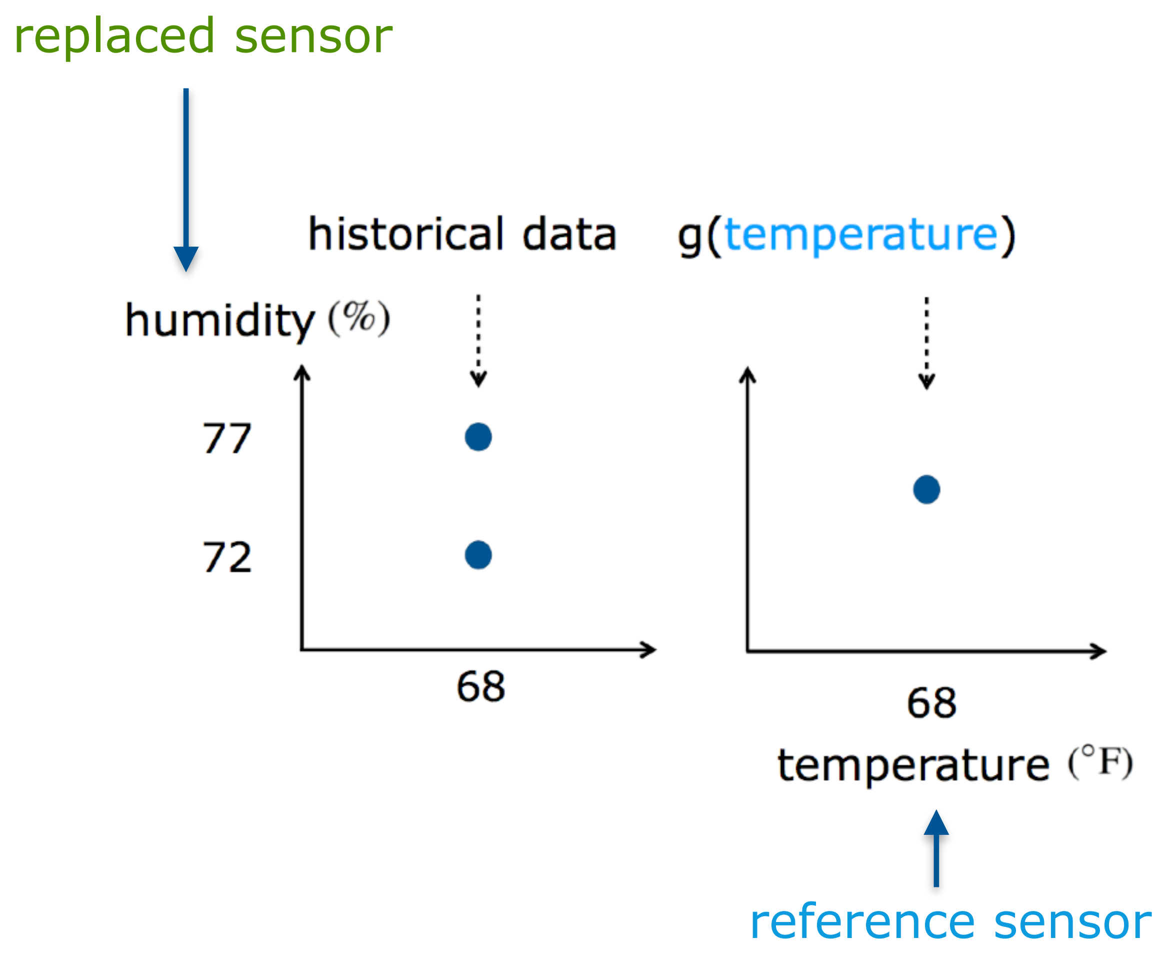
**Unexplored in previous work**



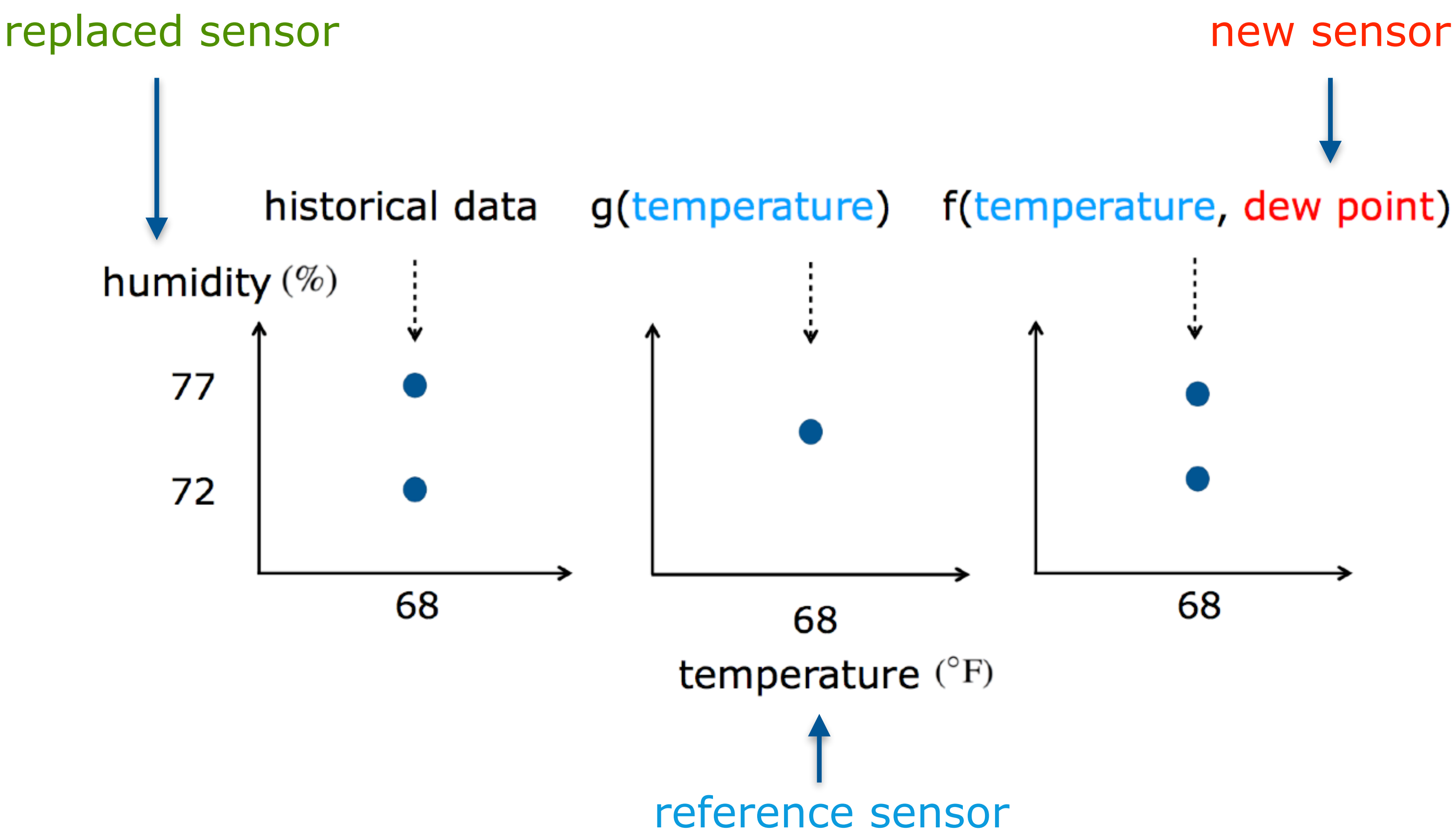

**Goal**: learning a **reconstruction function**:

f(reference sensors, new sensors) $\longrightarrow$ replaced sensors

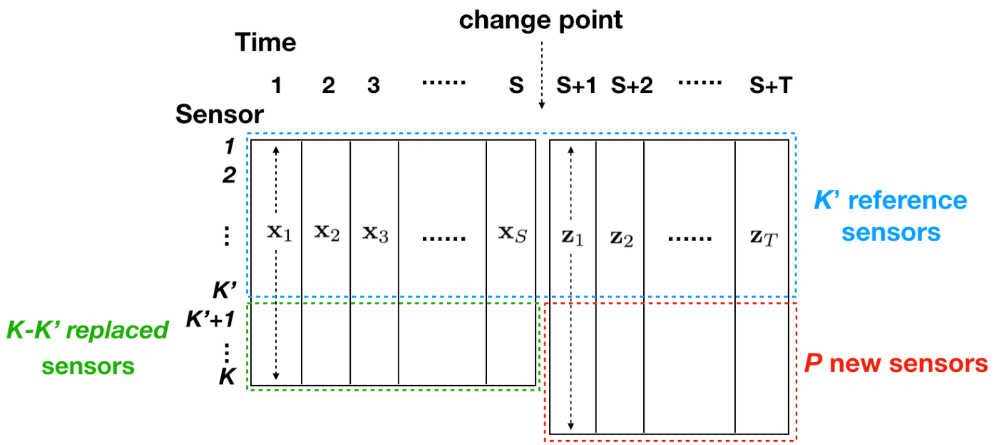
# Intuition of Exploiting New Sensors



# Intuition of Exploiting New Sensors



# Adaptation to Sensor Replacements

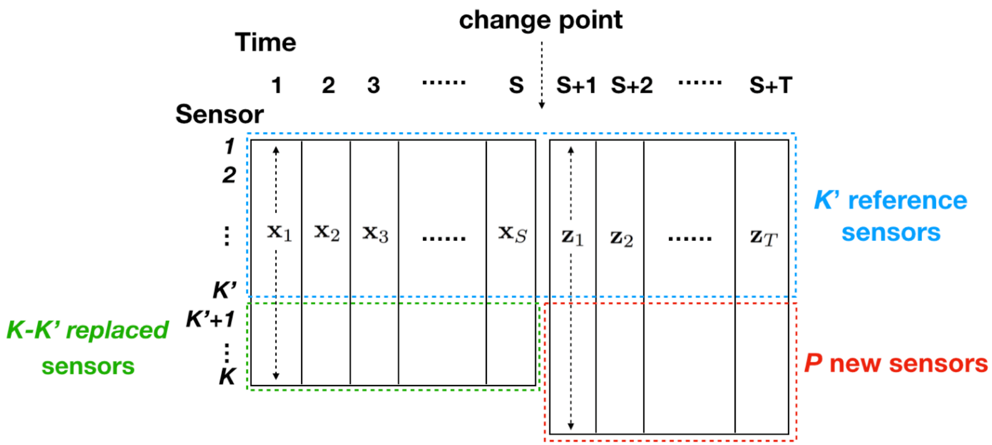



**Unexplored in previous work**

Reconstruction function:
f(reference sensors, new sensors) $\longrightarrow$ replaced sensors

**Challenge**: **no overlapping** between the replaced sensors and new sensors

# Adaptation to Sensor Replacements




**Unexplored in previous work**

Reconstruction function:
f(reference sensors, new sensors) ⟶ replaced sensors

**Challenge**: **no overlapping** between the replaced sensors and new sensors

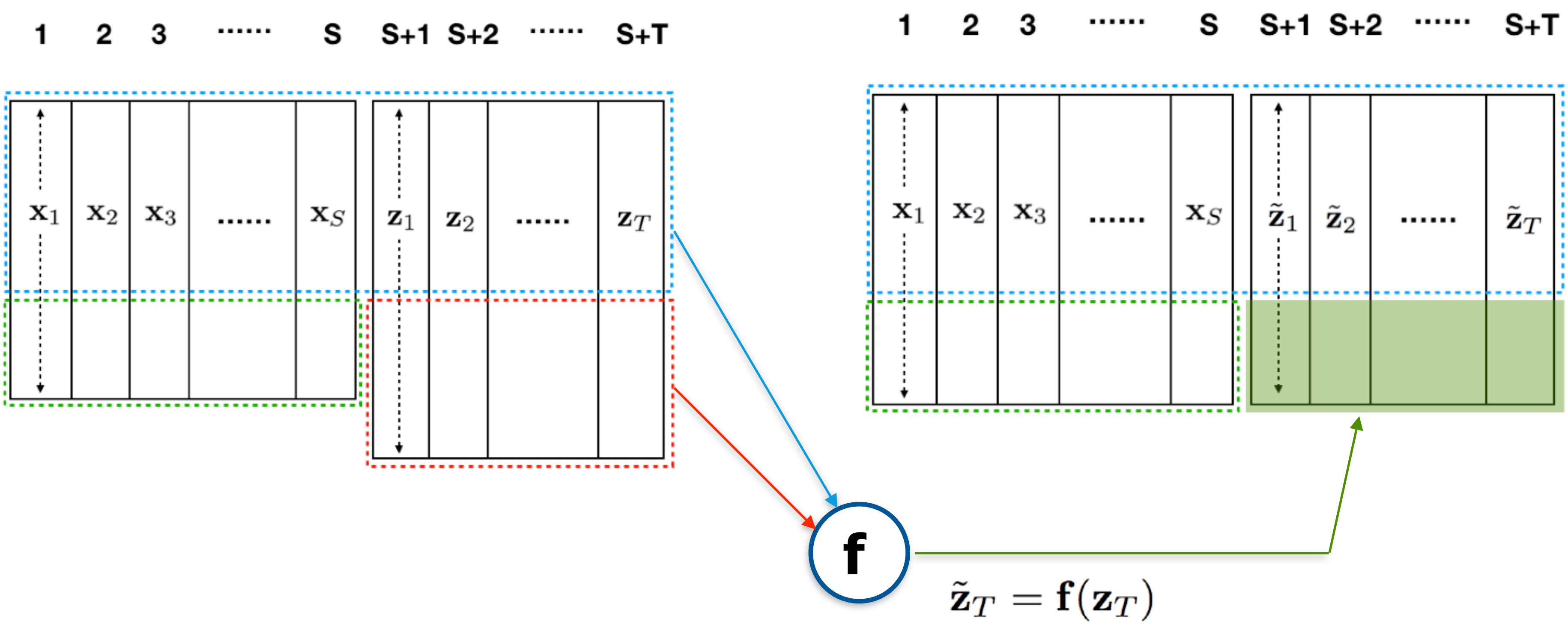
**Idea**: using the reference sensors as a bridge

**Assumption**:

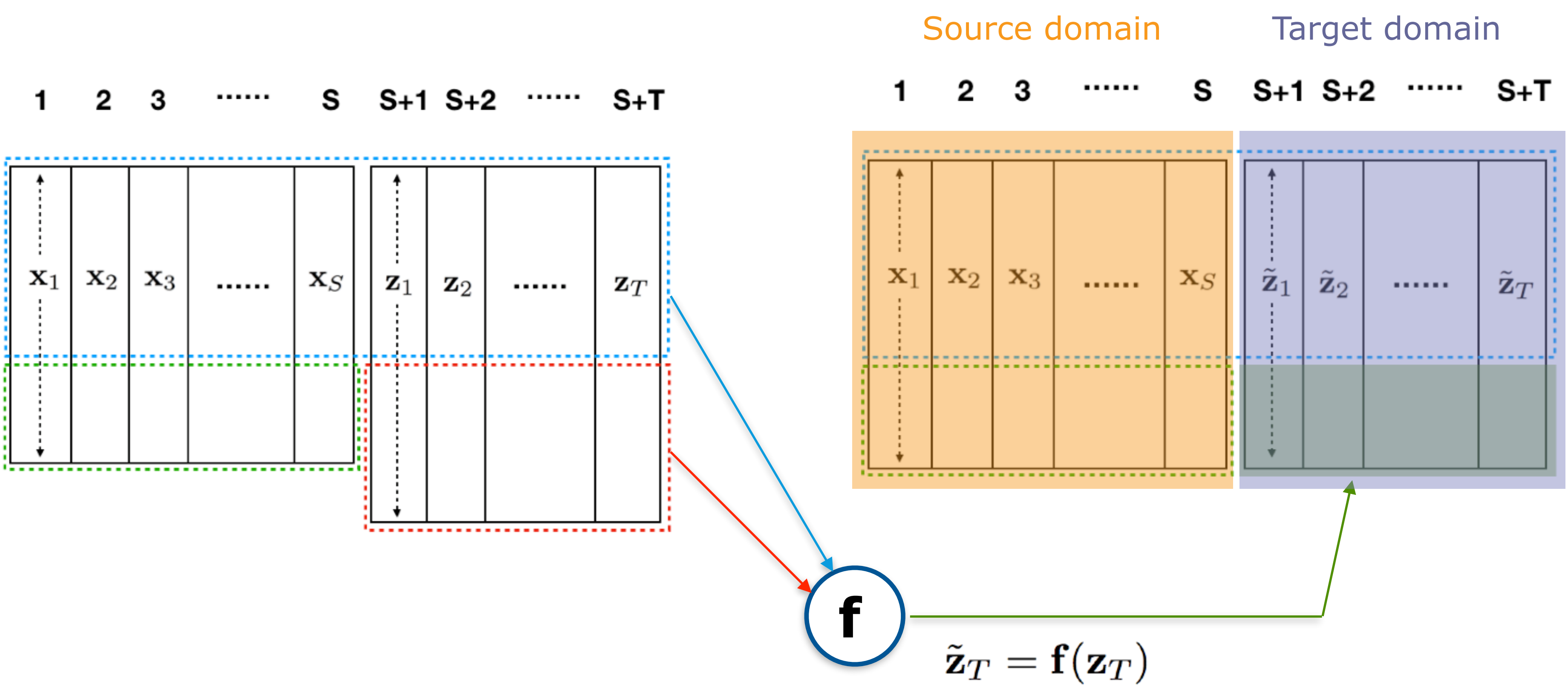
**1.** Sensor values from reference sensors are correlated with those from replaced sensors

**2.** Sensor values from reference sensors are correlated with those from new sensors

# Methodology of Our Approach

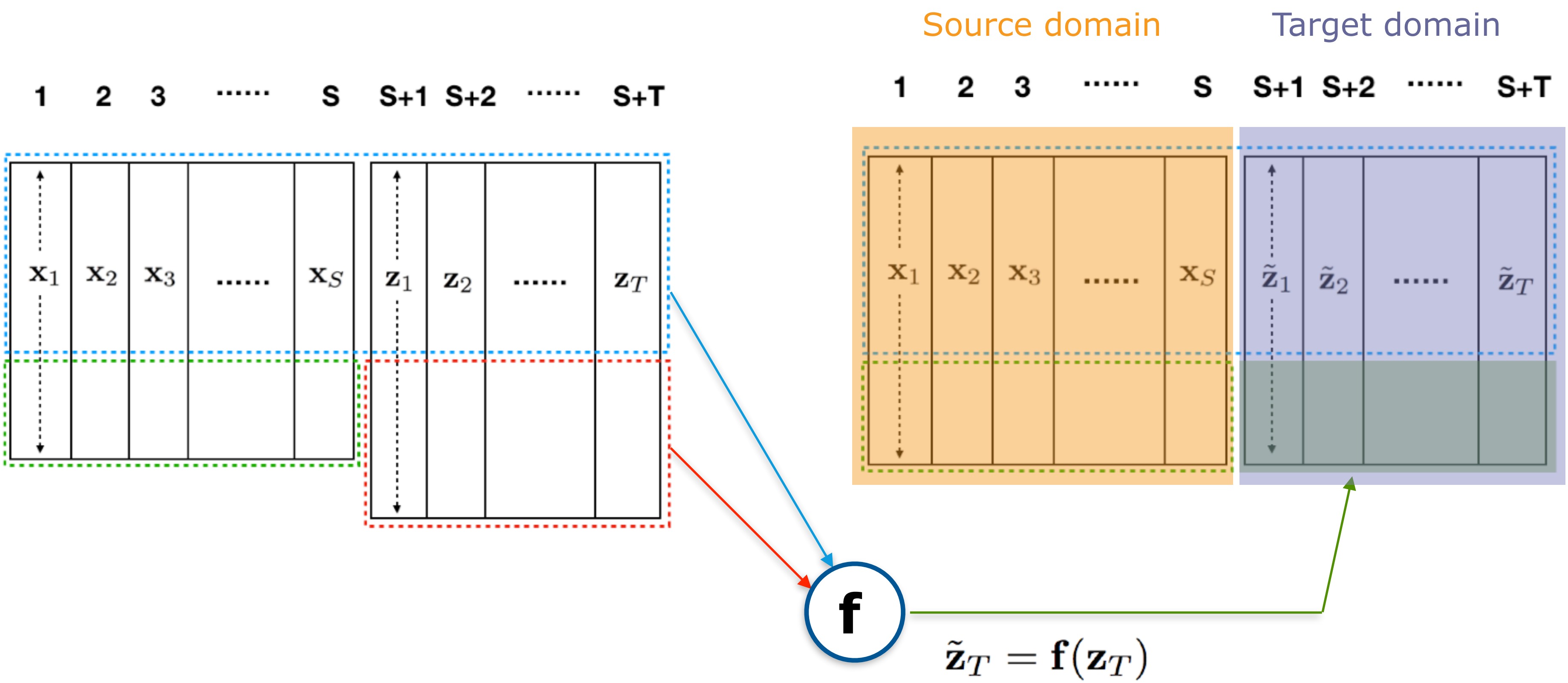


# Methodology of Our Approach



# Methodology of Our Approach

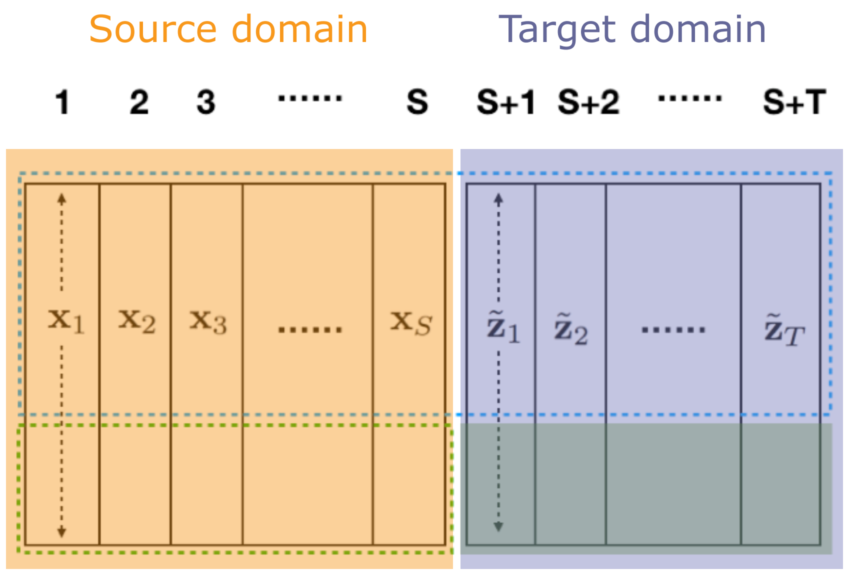
Samples in the two domains distribute similarly



# Methodology of Our Approach

Samples in the two domains distribute similarly

Two sets of samples mixed as much as possible




Minimize cross-domain *k*-nearest neighbor distances

$$\min_{\theta} \sum_{s} \sum_{t\in\mathcal{N}_{T}^{k}(s)} \mathcal{D}(\mathbf{x}_s, \tilde{\mathbf{z}}_t) + \sum_{t} \sum_{s\in\mathcal{N}_{S}^{k}(t)} \mathcal{D}(\tilde{\mathbf{z}}_t, \mathbf{x}_s)$$

$\mathbf{x}_s$'s *k* neighbors in the target domain

$\tilde{\mathbf{z}}_t$'s *k* neighbors in the source domain

# Formulation and Optimization

$$\min_{\boldsymbol{\Theta}} \sum_{s=1}^{S} \sum_{t \in \mathcal{N}_{\mathcal{T}}^{k}(s)} \mathcal{D}(\mathbf{x}_s, [\mathbf{z}_{t,1:K'}; \mathbf{f}_{\boldsymbol{\Theta}}(\mathbf{z}_t)]) + \sum_{t=1}^{T} \sum_{s \in \mathcal{N}_{\mathcal{S}}^{k}(t)} \mathcal{D}([\mathbf{z}_{t,1:K'}; \mathbf{f}_{\boldsymbol{\Theta}}(\mathbf{z}_t)], \mathbf{x}_s) + \lambda \|\boldsymbol{\Theta}\|_2^2$$

$$\mathbf{f}_{\boldsymbol{\Theta}}(\mathbf{z}) = \boldsymbol{\Theta}^{\mathrm{T}} \mathbf{h}(\mathbf{z})$$

regularization term

non-smooth in $\boldsymbol{\Theta}$, because neighbors are dependent on $\boldsymbol{\Theta}$

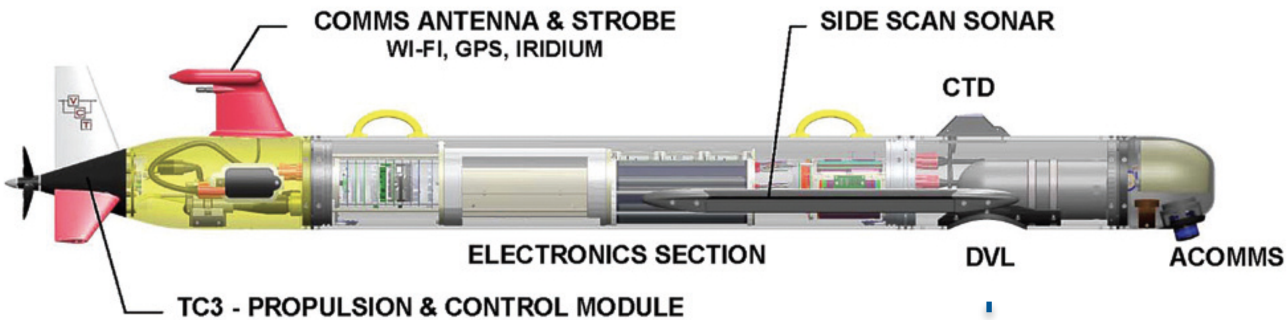
Alternating Optimization (EM-like algorithm):

- Fix $\boldsymbol{\Theta}$, update neighbors $\mathcal{N}_{\mathcal{T}}^{k}(s)$ and $\mathcal{N}_{\mathcal{S}}^{k}(t)$
- Fix neighbors $\mathcal{N}_{\mathcal{T}}^{k}(s)$ and $\mathcal{N}_{\mathcal{S}}^{k}(t)$, update $\boldsymbol{\Theta}$

# Results on UUV Data

A UUV travels from a starting point to an end point in a simulated environment

**Sensors**: propeller RPM, waterspeed, DVL (surge, heave, sway, pitch, roll, depth, heading)




surge, heave, sway, pitch, roll, depth, heading

# Results on UUV Data: Individual Sensor Replacement

**Replaced sensor**: surge/heave/sway
**New sensor**: biased version of surge/heave/sway
**Reference sensors**: remaining sensors

Directly using the new sensor

Adaptation using the reference sensors only

Adaptation using the reference sensors and predicted values of the new sensors

| Sensor | **Replace** | **Refer** | **Refer-Z** | **ASC** | Imp.(%) |
|---|---|---|---|---|---|
| surge | 2.47 | 0.66 | 0.58 | 0.47 | 18.9 |
| heave | 0.13 | 0.020 | 0.020 | 0.019 | 6.5 |
| sway | 2.31 | 0.74 | 0.72 | 0.71 | 1.1 |

Our approach

Reconstruction errors (RMSE) averaged over 20 simulated trips

**ASC** achieves an average improvement of 8.8% over the competing methods

# Results on UUV Data: Compound Sensor Replacement

**Replaced sensors**: all DVL sensors
**New sensor**: biased version of surge, heave and sway
**Reference sensors**: remaining sensors (propeller RPM, waterspeed)

Directly using the new sensor

Adaptation using the reference sensors only

Adaptation using the reference sensors and predicted values of the new sensors

| Sensor | **Replace** | **Refer** | **Refer-Z** | **ASC** | Imp.(%) |
|---|---|---|---|---|---|
| surge | 2.47 | 0.71 | 0.67 | 0.62 | 6.0 |
| heave | 0.094 | 0.026 | 0.026 | 0.024 | 3.4 |
| sway | 2.31 | 0.78 | 0.75 | 0.75 | -0.5 |

Our approach

Reconstruction errors (RMSE) averaged over 20 simulated trips

**ASC** achieves an average improvement of 3.0% over the competing methods

# Results on Weather Data

## Weather Underground Data

30 weather stations from 10 geographical clusters

**Sensors**: temperature, dew point, humidity, wind speed, wind gust, pressure

Compared to baselines, our approach **ASC** achieves

**6.4%** improvement for **individual sensor replacements**

**5.7%** improvement for **compound sensor replacements**

# Further Improvements

- **Dealing with many sensors**
  - Challenging due to more noise in sensor values
  - **Approach**: select a subset of the reference sensors and new sensors that are well correlated with the replaced sensors
- **Estimating adaptation quality**
  - Useful for upper-level software
  - **Approach**: produce an error interval instead of a single reconstruction value

# Related Work

- **Detecting Sensor Failures and Changes**
  - Change point detection [Aminikhanghahi and Cook '16] [Pimentel et al., '14]
    - Distribution-based [Kawahara and Sugiyama, '12] [Harchaoui et al., '09] [Yamanishi and Takeuchi, '02]
    - Reconstruction-based [Crook et al., '02] [Singh and Markou, '04] [Ide and Tsuda, '07] [Chatzigiannakis et al., '06]
    - Probabilistic [Adams and MacKay, '07] [Saatci et al., '10] [Dereszynski and Dietterich, '12] [Dietterich et al. '12]
    - Distance-based [Angiulli and Pizzuti, '02] [Bay and Schwabacher, '03] [Chawla and Sun, '06] [Keogh et al., '01] [Ide et al., '13] [Budalakoti et al., '06] [Chen et al., '15]

Most detection methods do not address how to automatically adapt to failures or changes

- **Reconstruction of Sensor Values**
  - Some probabilistic methods [Dereszynski and Dietterich, '12] [Dietterich et al. '12] can be used to reconstruct changed sensor, but cannot leverage new sensors
  - Our earlier work [Yuan and Craig, '17] can only exploit a single new sensor

Our approach can adapt to multiple new sensors, which are not possible by existing approaches

# Conclusion

- A novel problem of automatically adapting to sensor replacements
  - In the context of building long-lived, survivable software
- A machine learning approach capable of
  - Exploiting new sensors
  - Scaling to many sensors
  - Estimating the adaptation quality

  - Supported by empirical study in the UUV and weather domains