Smart EPU
Workshop on Smart Data Collection for CryoEM, 6-7 April 2022

Fanis Grollios
Sr. Software Product Manager

Lingbo Yu
Product Marketing Manager
Smart EPU—How did we get here?

**EPU 1.0**
(2011-2018)
Enabling SPA

**EPU 2.0**
(2018-2022)
Easier and more efficient SPA

**EPU 3.0**
Automated SPA

Increasing ease of use and microscope efficiency
Introducing the Smart EPU ecosystem
Smart EPU

A platform to enhance efficiency and ease of use for Cryo-EM

EPU Quality Monitor

EPU Decision Service with Open API

Increase efficiency / Reduce Expertise

ThermoFisher

“Plug-in” Development

Community

“Plug-in” Development
Smart Plug-ins

Focus Stabilization
Stage Waiting Optimization
Skip Gridsquare
Automatic Foil Hole Selection

Smart Plugins use real-time data to make *automated* decisions, making the workflow more efficient and reducing the need for prior expertise.
Smart EPU in action
Smart EPU: monitoring key parameters
Smart plugin #1: automatically adjust focus
Smart Focus results

Faster and more consistent focus values measured

- EPU Auto-focus
- Smart Focus

**Data Quality**
Focusing becomes more accurate as it is based on CTF fits from acquired images.

**Efficiency**
Minimization of beam-tilt-based autofocus executions improves throughput

**Ease of Use**
Smart Focus routine intervenes and re-adjusts user settings on its own

*600 acquired images*
Smart plugin #2: automatically skip bad areas

Skip grid squares with CTF resolution estimation above 6 Å

Benefits

Data Quality
Areas that lead to suboptimal data are automatically excluded

Efficiency
The fraction of useable data is increased, so fewer images are needed.

Ease of Use
No need to pre-select grid squares carefully as the selection will be optimized on the fly.
Skip Gridsquare in Action

Square1

Square2

Square3
AI Foil Hole Selection

**Training**
Grid square images are used to train a Neural Network to identify good/bad foil holes

**Inference**
The network can be used by EPU to determine the selection of holes to be acquired

---

**Benefits**

**Data Quality**
Areas that are predicted to give sub-optimal data are automatically excluded

**Efficiency**
Less time is needed to exclude bad foil holes by hand. Smart EPU automatically creates an efficient set-up

**Ease of Use**
No expertise needed to identify bad foil holes.
Set up using Smart Algorithms

Algorithm automatically excludes sub-optimal areas
EPU Multigrid

Combine easy set-up with EPU Multigrid

- Automatic atlas acquisitions
- Compare grids
- Square categorization

Screening

- Easy preparation in advance
- Grid exchange and alignment
- Stopping conditions per session

Multigrid queue

- Best image quality using Falcon4
- High throughput with FFI and AFIS
- CFEG and Selectris integration

Automated acquisition

Assembly of recombinant tau filaments identical to those of Alzheimer's disease and chronic traumatic encephalopathy. Lövestam, et al., eLife 2022;11:e76494 DOI: 10.7554/eLife.76494
Community Plugins / Open API

- **Method development**
  Enables users to develop specific plugins for their needs

- **Expedite plugin development**
  Enables faster innovation of new automation

Our robust API enables the community to be part of the ecosystem.
An API to retrieve data and drive EPU

# 1. Get area currently processed by EPU
app_state = requests.get("http://decision-service:5000/CurrentApplicationState").json()
current_area_id = app_state["areaId"]  # we assume here that it's a foil hole

# 2. Retrieve relevant motion correction results to calculate new stage waiting time
grid_square_id = requests.get(f"http://decision-service:5000/Area/{current_area_id}").json()["parentId"]
motion_correction_results = requests.get("http://decision-service:5000/AlgorithmResults", params={"parentAreaId": grid_square_id, "name": "motioncorrection"}, ).json()

# 3. Compute new stage waiting time based on motion correction results
new_stage_waiting_time = ...

# 4. Register a new stage settling decision for future areas ("...")
requests.post("http://decision-service:5000/Decision", json={"areaId": ..., "decisionType": "stageWaitingTime", "decisionValue": new_stage_waiting_time, "decidedBy": "smart algorithm"})
Community Algorithms: 2 Options

- Athena DB
- EQM

Decision Service with Open API
Complete DIY
Build your own Algorithms

Use own solution for processing/managing data
- Copy data from our domain with own means
- Run all processing on your own infrastructure

Communicate with decision service API to feedback optimizations to EPU

Decision Service with Open API

EPU
Plugin DIY
Leverage Athena and EQM as well

Use Athena/EQM for processing/managing data
- Retrieve pre-processed data and results through API

Deploy plugins on your own infrastructure

Communicate with decision service API to feedback optimizations to EPU
Community Algorithms: What shall we Provide?

Enable the Community to build their own Algorithms

Documentation and Support on API

Example algorithms

What else would you need?
Do we need a standard API?
Questions

EPU@Thermofisher.com