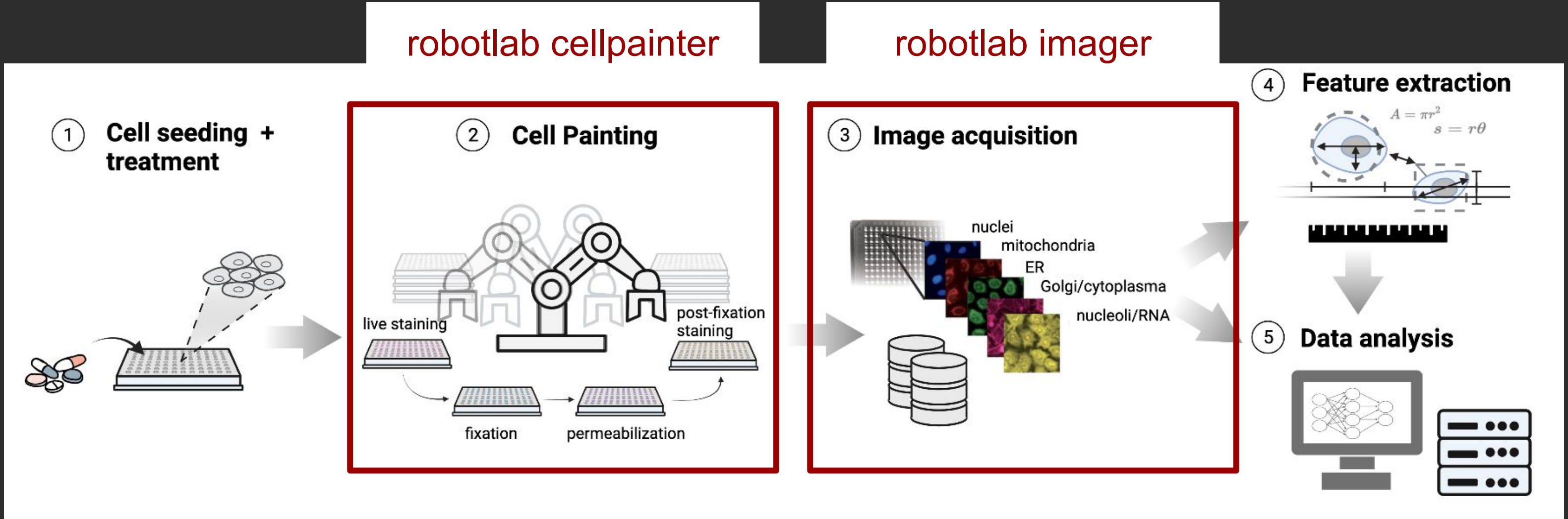
pharmbio robotlab

Dan Rosén, 2023-08-30

High-content cell profiling



robotlab cellpainter

robotlab imager

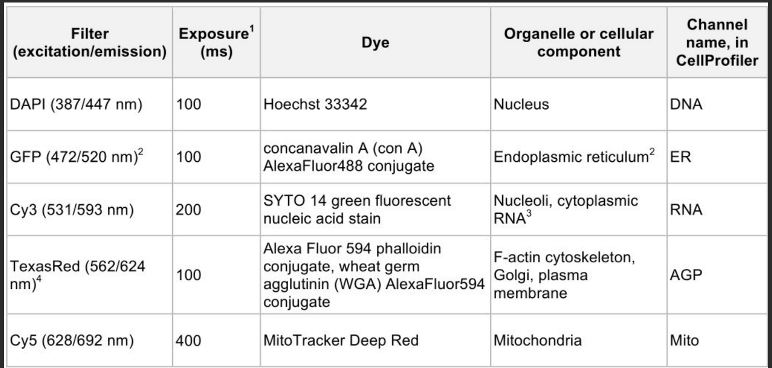
Cell painting protocol

Each plate goes through:

- 5 washes (70s-135s)
- 4 dispenses (20s-30s)
- 4 incubation (20 min)

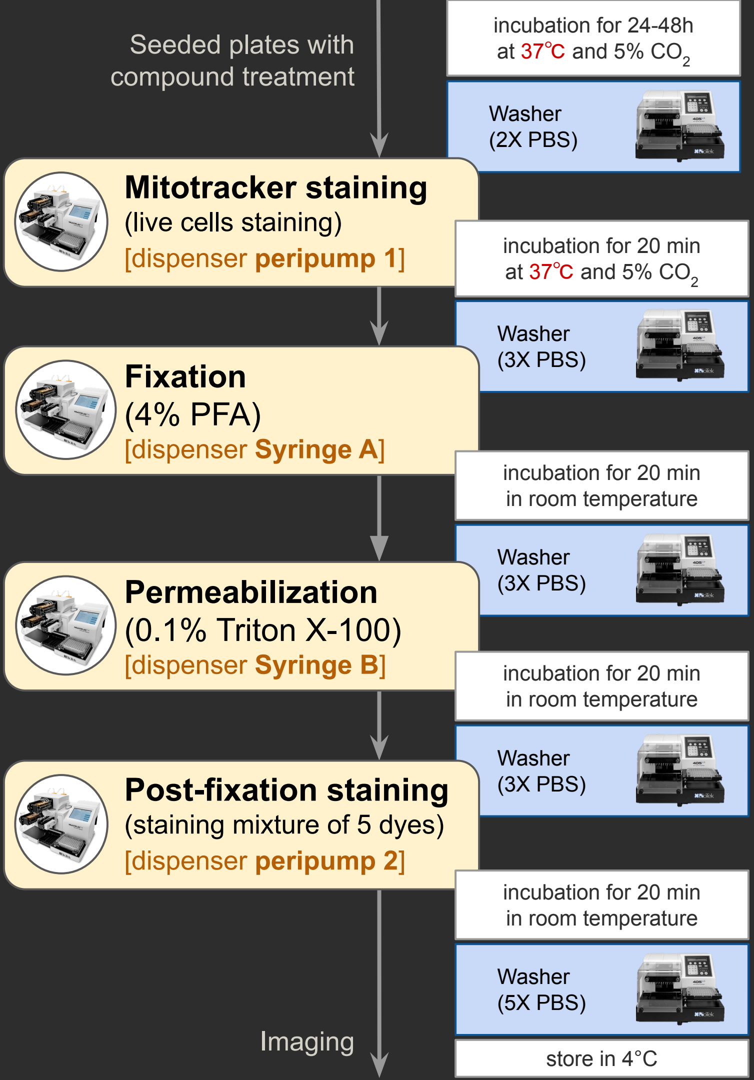
Lid on when incubating

Lid off when using washer and dispenser



Filter (excitation/emission)	Exposure[1] (ms)	Dye	Organelle or cellular component	Channel name, in CellProfiler
DAPI (387/447 nm)	100	Hoechst 33342	Nucleus	DNA
GFP (472/520 nm)[2]	100	concanavalin A (con A) AlexaFluor488 conjugate	Endoplasmic reticulum[2]	ER
Cy3 (531/593 nm)	200	SYTO 14 green fluorescent nucleic acid stain	Nucleoli, cytoplasmic RNA[3]	RNA
TexasRed (562/624 nm)[4]	100	Alexa Fluor 594 phalloidin conjugate, wheat germ agglutinin (WGA) AlexaFluor594 conjugate	F-actin cytoskeleton, Golgi, plasma membrane	AGP
Cy5 (628/692 nm)	400	MitoTracker Deep Red	Mitochondria	Mito

Cell Painting, an image-based assay for morphological profiling (Bray et al, 2016)



Seeded plates with compound treatment

incubation for 24-48h at 37°C and 5% CO$_2$

Washer (2X PBS)

Mitotracker staining (live cells staining) [dispenser **peripump 1**]

incubation for 20 min at 37°C and 5% CO$_2$

Washer (3X PBS)

Fixation (4% PFA) [dispenser **Syringe A**]

incubation for 20 min in room temperature

Washer (3X PBS)

Permeabilization (0.1% Triton X-100) [dispenser **Syringe B**]

incubation for 20 min in room temperature

Washer (3X PBS)

Post-fixation staining (staining mixture of 5 dyes) [dispenser **peripump 2**]

incubation for 20 min in room temperature

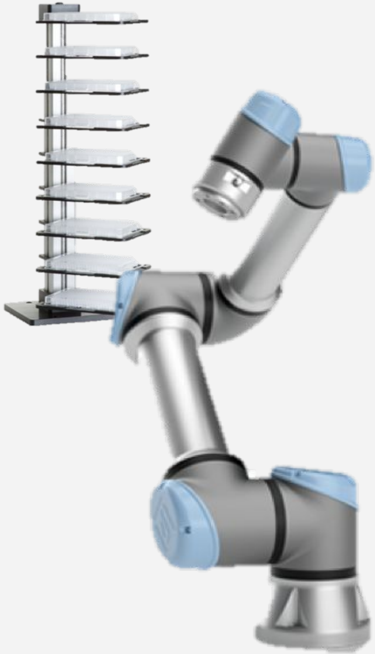
Washer (5X PBS)

Imaging

store in 4°C

Automated cell painting

37C incubator
(Liconic)



Robot arm (UR)

Biotek dispenser

Biotek washer

Components `github.com/pharmbio/robotlab`

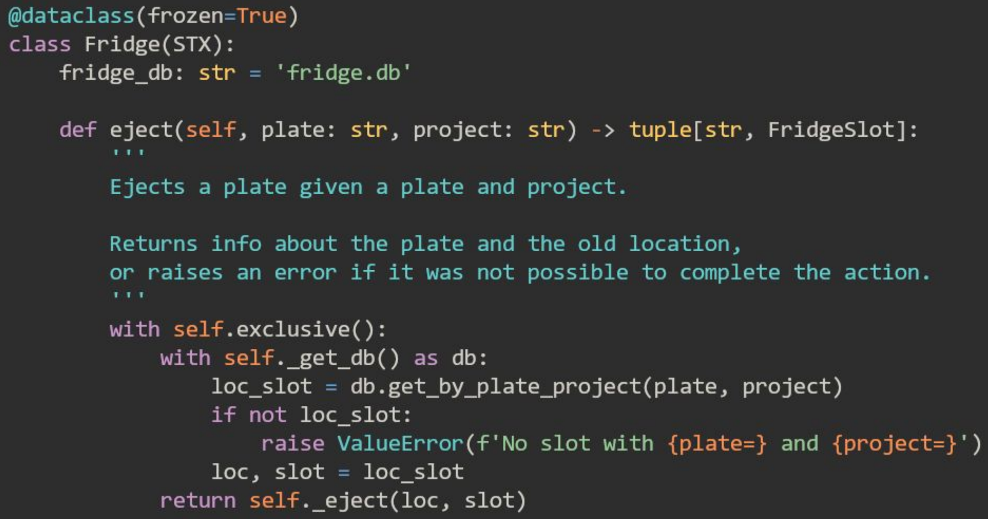
- Machine communication `labrobots/`
  - HTTP server (flask) wrapping lower-level communication (COM-ports, subprocesses, ...)
  - Type-safe python wrapper
  - Developers can access using curl
- Robotarm moves `cellpainter/{moves_gui.py,movelists/*jsonl}`
  - Custom UI
  - Interpolate between hotel positions
  - Moves cut up into smaller pieces using code
- Scheduling `cellpainter/{commands,optimize,protocol}.py`
  - DSL for machine execution and time checkpointing.
  - SMT solver Z3 to solve waiting delays.
- Front end web browser UI `cellpainter/gui/`
  - Generated server-side from python
  - CLI also present but never used

Fridge communication

```
dan@devserver:~$ curl 10.10.0.97:5050
{
  "http://10.10.0.97:5050/echo": "Echo()",
  "http://10.10.0.97:5050/git": "Git()",
  "http://10.10.0.97:5050/fridge":
    "Fridge(id='STX', host='localhost', port=3333, mode='execute',
       current_climate=Cell(value={'temp': 4.5, 'humid': 83.6, 'co2': 0.0, 'n2': 240.23}),
       fridge_db='fridge.db')",
  "http://10.10.0.97:5050/barcode": "BarcodeReader(com_port='COM3', current_barcode=Cell(value='P102813'))",
  "http://10.10.0.97:5050/io.sql": "Download the IO database as sqlite dump",
  "http://10.10.0.97:5050/io.db": "Download the IO database in binary sqlite",
  "http://10.10.0.97:5050/tail": "Show last 10 lines from the IO database",
  "http://10.10.0.97:5050/tail/<N>": "Show last N lines from the IO database"
}
```

```
dan@devserver:~$ curl 10.10.0.97:5050/tail
280319  2023-08-29 13:31:03.793  fridge   48879  eject(plate='P102813', project='SSS-val')
280320  2023-08-29 13:31:03.808  fridge   48879  stx.write(b'STX2ServiceMovePlate(STX,2,3,2,1,1,STX,1,0,0,1,1)\r')
280321  2023-08-29 13:31:35.048  fridge   48879  stx.read() = b'1\r\n'
280322  2023-08-29 13:31:35.115  fridge   48879  31312.0ms eject(plate='P102813', project='SSS-val')
280323  2023-08-29 13:31:35.124  fridge   48879  return ('3x2', {'plate': 'P102813', 'project': 'SSS-val'})
```

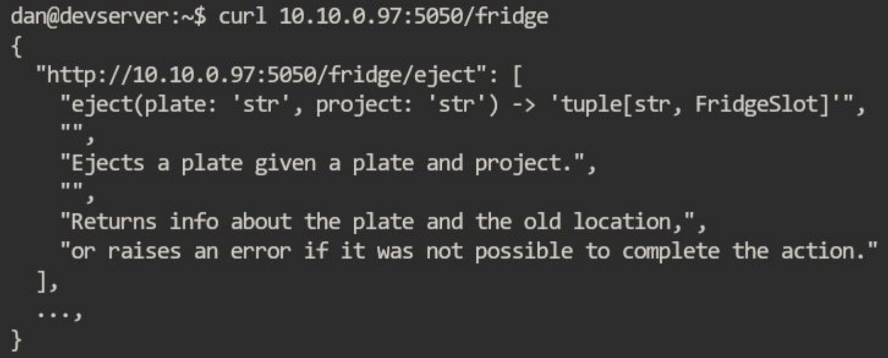
Fridge communication



```
@dataclass(frozen=True)
class Fridge(STX):
    fridge_db: str = 'fridge.db'

    def eject(self, plate: str, project: str) -> tuple[str, FridgeSlot]:
        '''
        Ejects a plate given a plate and project.

        Returns info about the plate and the old location,
        or raises an error if it was not possible to complete the action.
        '''
        with self.exclusive():
            with self._get_db() as db:
                loc_slot = db.get_by_plate_project(plate, project)
                if not loc_slot:
                    raise ValueError(f'No slot with {plate=} and {project=}')
                loc, slot = loc_slot
            return self._eject(loc, slot)
```

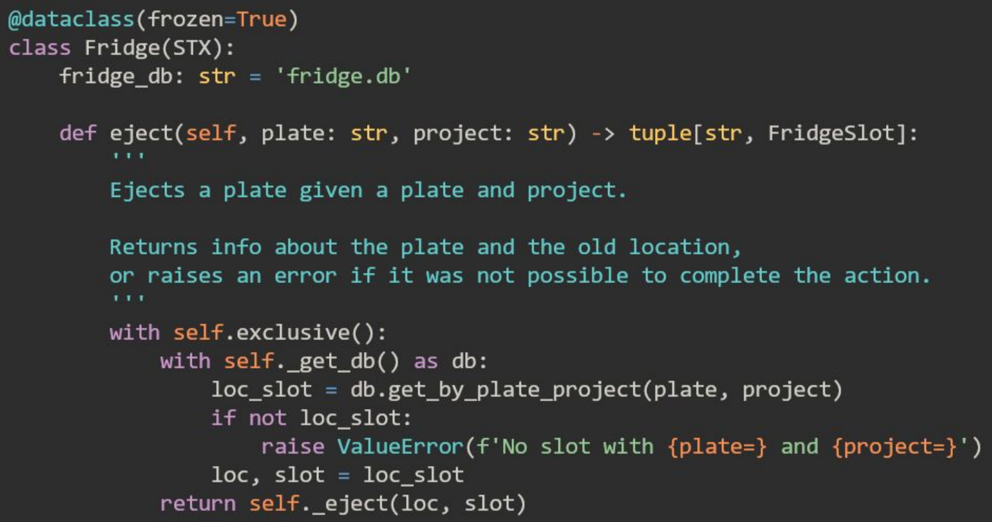


```
dan@devserver:~$ curl 10.10.0.97:5050/fridge
{
  "http://10.10.0.97:5050/fridge/eject": [
    "eject(plate: 'str', project: 'str') -> 'tuple[str, FridgeSlot]'",
    "",
    "Ejects a plate given a plate and project.",
    "",
    "Returns info about the plate and the old location,",
    "or raises an error if it was not possible to complete the action."
  ],
  ...,
}
```

HTTP endpoints generated from the class

```
curl …/fridge/eject/P001337/protac
curl -X POST …/fridge/eject -d ‘{
  “plate”:”P1337”,”project”:”protac”}’
```

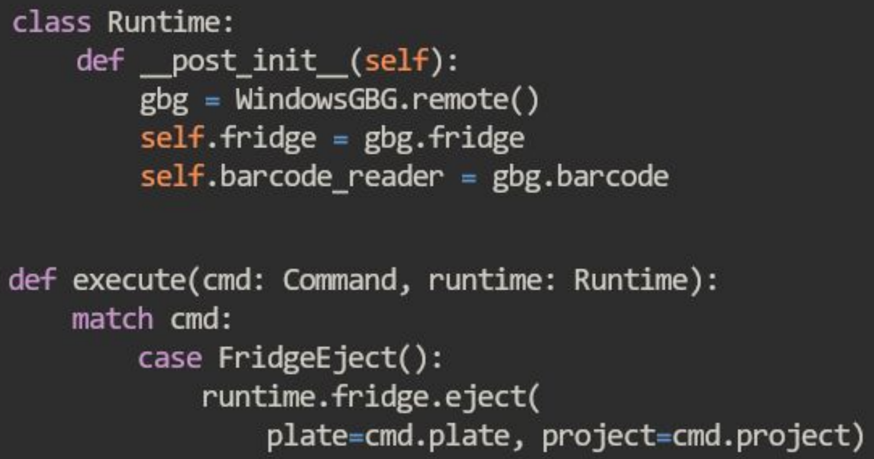
Fridge communication



```python
@dataclass(frozen=True)
class Fridge(STX):
    fridge_db: str = 'fridge.db'

    def eject(self, plate: str, project: str) -> tuple[str, FridgeSlot]:
        '''
        Ejects a plate given a plate and project.

        Returns info about the plate and the old location,
        or raises an error if it was not possible to complete the action.
        '''
        with self.exclusive():
            with self._get_db() as db:
                loc_slot = db.get_by_plate_project(plate, project)
                if not loc_slot:
                    raise ValueError(f'No slot with {plate=} and {project=}')
                loc, slot = loc_slot
            return self._eject(loc, slot)
```

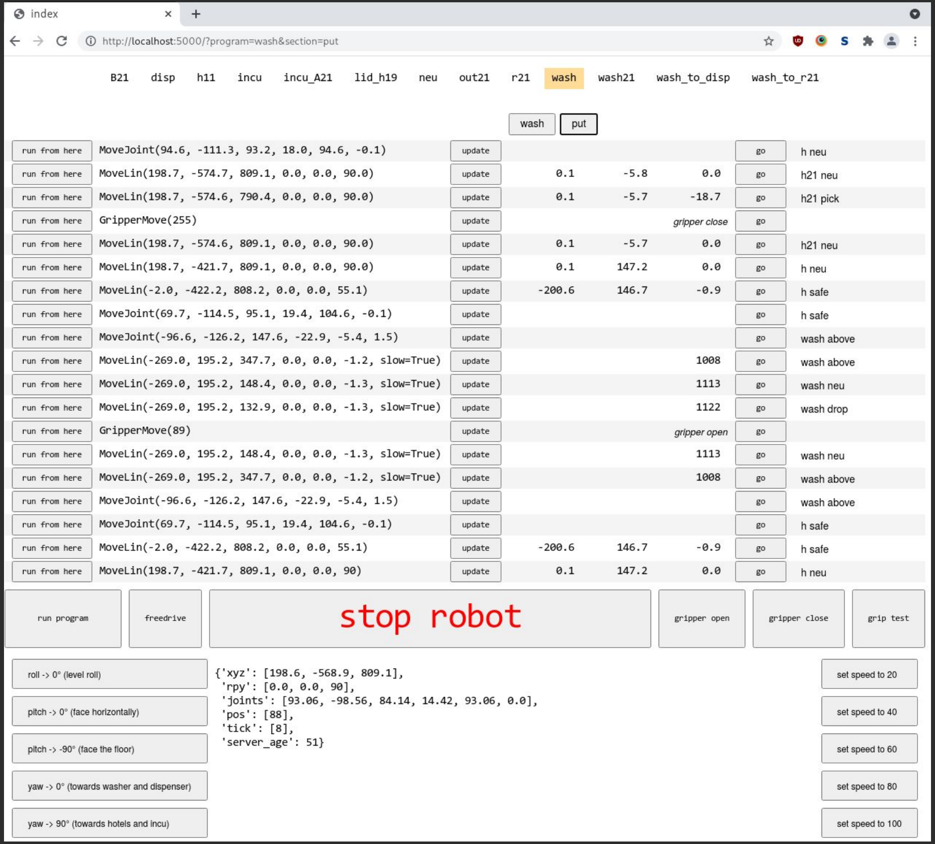


```python
class Runtime:
    def __post_init__(self):
        gbg = WindowsGBG.remote()
        self.fridge = gbg.fridge
        self.barcode_reader = gbg.barcode


def execute(cmd: Command, runtime: Runtime):
    match cmd:
        case FridgeEject():
            runtime.fridge.eject(
                plate=cmd.plate, project=cmd.project)
```

Type-safe communication:
Type-checker thinks `runtime.fridge` is a `Fridge`, but it is a proxy object that does RPC over HTTP.

GUI for creating robot arm moves

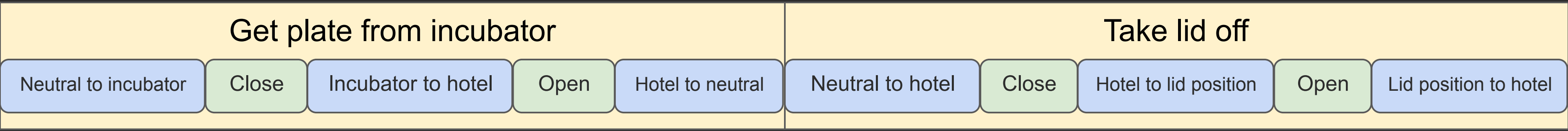


run from here	MoveJoint(94.6, -111.3, 93.2, 18.0, 94.6, -0.1)	update				go	h neu
run from here	MoveLin(198.7, -574.7, 809.1, 0.0, 0.0, 90.0)	update	0.1	-5.8	0.0	go	h21 neu
run from here	MoveLin(198.7, -574.6, 790.4, 0.0, 0.0, 90.0)	update	0.1	-5.7	-18.7	go	h21 pick
run from here	GripperMove(255)	update			*gripper close*	go	
run from here	MoveLin(198.7, -574.6, 809.1, 0.0, 0.0, 90.0)	update	0.1	-5.7	0.0	go	h21 neu
run from here	MoveLin(198.7, -421.7, 809.1, 0.0, 0.0, 90.0)	update	0.1	147.2	0.0	go	h neu
run from here	MoveLin(-2.0, -422.2, 808.2, 0.0, 0.0, 55.1)	update	-200.6	146.7	-0.9	go	h safe
run from here	MoveJoint(69.7, -114.5, 95.1, 19.4, 104.6, -0.1)	update				go	h safe
run from here	MoveJoint(-96.6, -126.2, 147.6, -22.9, -5.4, 1.5)	update				go	wash above
run from here	MoveLin(-269.0, 195.2, 347.7, 0.0, 0.0, -1.2, slow=True)	update			1008	go	wash above
run from here	MoveLin(-269.0, 195.2, 148.4, 0.0, 0.0, -1.3, slow=True)	update			1113	go	wash neu
run from here	MoveLin(-269.0, 195.2, 132.9, 0.0, 0.0, -1.3, slow=True)	update			1122	go	wash drop
run from here	GripperMove(89)	update			*gripper open*	go	
run from here	MoveLin(-269.0, 195.2, 148.4, 0.0, 0.0, -1.3, slow=True)	update			1113	go	wash neu
run from here	MoveLin(-269.0, 195.2, 347.7, 0.0, 0.0, -1.2, slow=True)	update			1008	go	wash above
run from here	MoveJoint(-96.6, -126.2, 147.6, -22.9, -5.4, 1.5)	update				go	wash above
run from here	MoveJoint(69.7, -114.5, 95.1, 19.4, 104.6, -0.1)	update				go	h safe
run from here	MoveLin(-2.0, -422.2, 808.2, 0.0, 0.0, 55.1)	update	-200.6	146.7	-0.9	go	h safe
run from here	MoveLin(198.7, -421.7, 809.1, 0.0, 0.0, 90)	update	0.1	147.2	0.0	go	h neu

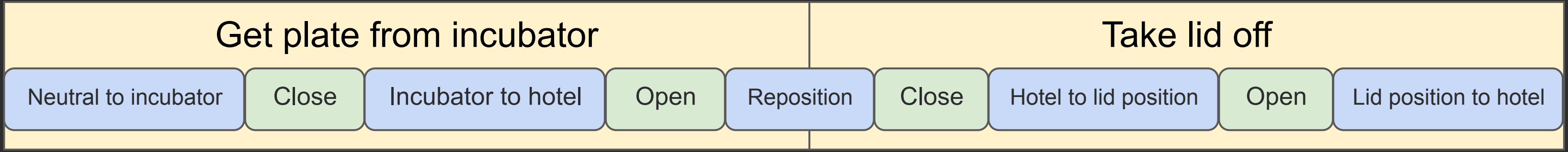
```
{'xyz': [198.6, -568.9, 809.1],
 'rpy': [0.0, 0.0, 90],
 'joints': [93.06, -98.56, 84.14, 14.42, 93.06, 0.0],
 'pos': [88],
 'tick': [8],
 'server_age': 51}
```

- Written for the Universal Robots robot, later ported to the PreciseFlex robot

Designing and sequencing robot arm moves



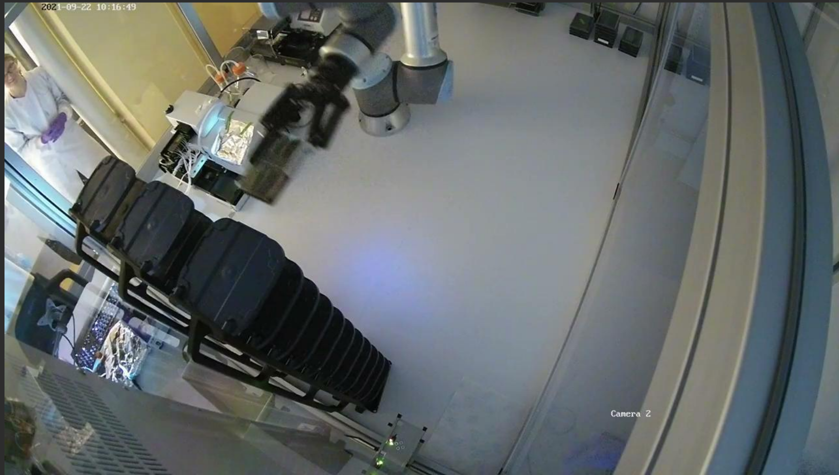
Get plate from incubator					Take lid off				
Neutral to incubator	Close	Incubator to hotel	Open	Hotel to neutral	Neutral to hotel	Close	Hotel to lid position	Open	Lid position to hotel



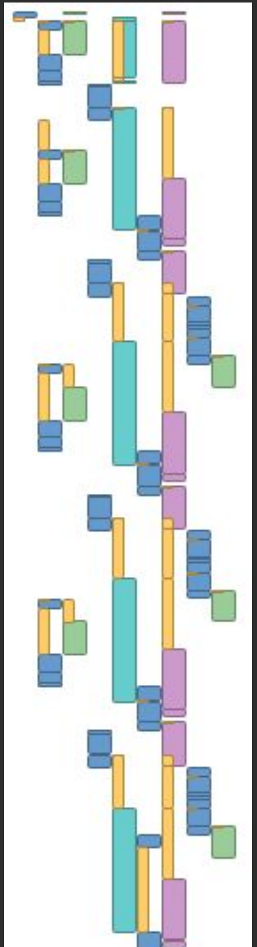
Get plate from incubator					Take lid off			
Neutral to incubator	Close	Incubator to hotel	Open	Reposition	Close	Hotel to lid position	Open	Lid position to hotel

- We use a neutral position by the hotel as a reference point for all moves
- When sequencing two moves we can remove redundant returns to the neutral position

- We generate moves to all hotel positions by offsetting a chosen subset of the moves with the hotel separation distance



Expressing the protocol in code and scheduling it



The protocol is expressed in a domain-specific language (embedded in python):

```
cmd ::=    RunRobotarm(program_name)
      |    RunDispenser(protocol_name)
      |    Checkpoint(checkpoint_name)
      |    WaitForCheckpoint(checkpoint_name, plus_seconds)
      |    Fork(cmd)
      |    Sequence(cmd, cmd)
```

The program can be executed in three ways:

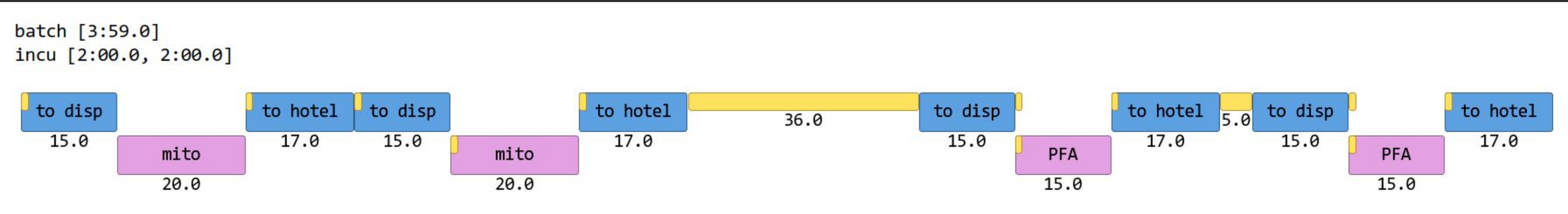
	1. Live	2. Simulation	3. Symbolically
Time:	Wall time	Simulated	Symbolic
Machines:	Execute	Advance time	Advance time
Purpose:	Side effects	Testing	Scheduling waits

Solve and optimize using off-the shelf SMT-solver (Z3)

Example: cell painting a batch of two plates

Example: cell painting a batch of two plates

robot arm
dispenser
delay



```
batch [3:59.0]
incu [2:00.0, 2:00.0]
```

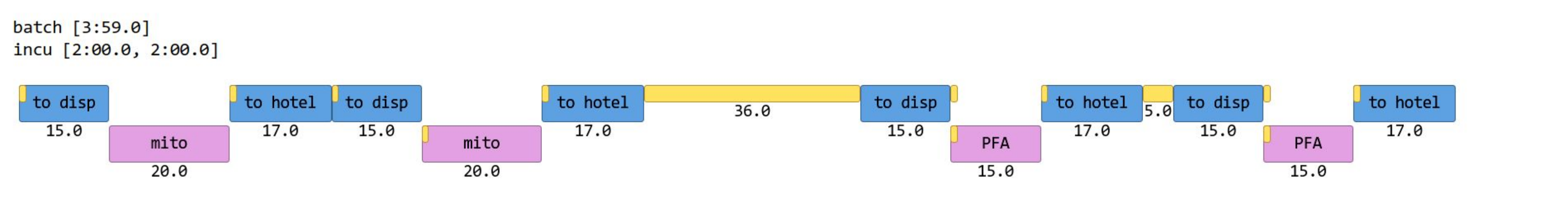
Protocol expressed in our Domain-Specific Language (DSL) embedded in Python.

```
cmd ::=  RunRobotarm(program_name)
     |   RunDispenser(protocol_name)
     |   Checkpoint(checkpoint_name)
     |   WaitForCheckpoint(checkpoint_name, plus_seconds)
     |   Fork(cmd)
     |   Sequence(cmd, cmd)
```

Scheduling by translation to SMT (fragment: Quantifier-Free Linear Arithmetic).

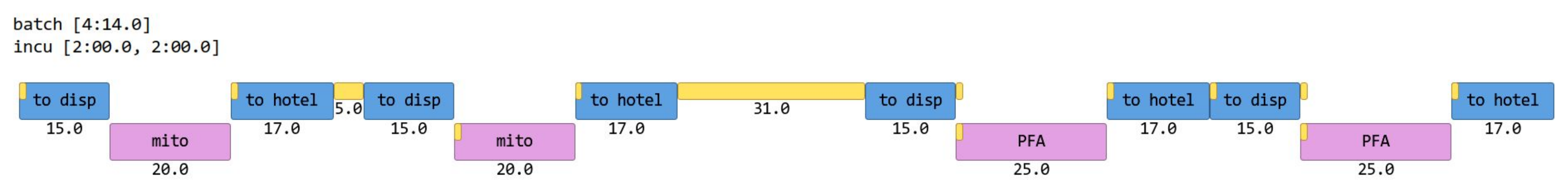
Example non-linearity

Case |PFA| < |Mito|:



```
batch [3:59.0]
incu [2:00.0, 2:00.0]
```

Case |PFA| > |Mito|:



```
batch [4:14.0]
incu [2:00.0, 2:00.0]
```

The delays have to be allocated differently depending on which case we are in. Formally, this means that the optimization problem is not linear.

Expressing complex protocols

Four plates in the MitoTracker cycle of the cell painting protocol.

robot arm

dispenser

incubator

washer

delay