Update on grading automation

Progress Mar 31 - Apr 4

- 3 version iterations for the processing program: v10 → v13
- 2 version iterations for the master processing program: v3 → v5
- 2 version iterations for JSON checking and uploading: v1 → v3
- With help from Jay, resolved naming and radius issues in the Keyence-side program. Now able to correct keyence program problem independently.
- With Jay and Yi-Mu, identified that Keyence-approved center holes fails the Go/No-Go gauge test often, manually inspected orange-grade plates: deburred and expanded as needed
- Provided assistance with cleaning and measuring tasks during staff shortages

20250401KeyenceIM-8030T.csv(raw output)

Program name	0:Flatness R	0:Flatness L	1:Width edge 0	1:Width corner 0	1:Radius 0	1:Radius 1	
Design value	0	0	166.94	94.07	83.88	74.81	
Upper Limit	0.2	0.2	0.2	0.2	0.1	0.1	.....
Lower Limit	-0.2	-0.2	-0.12	-0.2	-0.1	-0.06	
CuW_LD_Right_Combined	0.033	0.031	166.987	94.118	83.915	74.833	
CuW_LD_Right_Combined	0.089	0.108	166.958	94.134	83.94	74.852	
CuW_LD_Right_Combined	0.104	0.125	166.956	94.037	83.858	74.813	
CuW_LD_Right_Combined	0.052	0.067	166.947	94.041	83.849	74.799	
		.....					

After each measurement(for 1 geom, 1 batch, 1 day), a result is automatically generated along with a .bi3 extension file (checking with SQLite)

20250401KeyenceIM-8030T.csv(raw output)

mask.py

```
mask = {
    '1:Width edge 0': (166.94, 0.1, 0.1, 0.2, 0.12),        '1:Width corner 0': (94.07, 0.1, 0.1, 0.2, 0.2),
    '1:Radius 0': (83.88, 0.05, 0.05, 0.1, 0.06),           '1:Radius 1': (74.81, 0.05, 0.05, 0.1, 0.06),
    '1:Radius 2': (83.47, 0.05, 0.05, 0.1, 0.06),           '1:Radius 3': (10.19, 0.05, 0.05, 0.1, 0.06),
    '1:Radius 4': (83.47, 0.05, 0.05, 0.1, 0.06),           '1:Radius 5': (74.81, 0.05, 0.05, 0.1, 0.06),
    '1:Distance from hole to slot': (65.0, 0.1, 0.1, 0.2, 0.2), '1:Diameter hole': (3.05, 0.025, 0.025, 0.05,
0.05),
    '1:Length of slot': (3.55, 0.1, 0.1, 0.2, 0.2),         '1:Width of slot': (3.05, 0.025, 0.025, 0.05, 0.05),
    '1:Notch 0': (1.0, 0.1, 0.1, 0.2, 0.2),                '1:Notch 1': (1.0, 0.1, 0.1, 0.2, 0.2),
    '1:Notch 2': (1.0, 0.1, 0.1, 0.2, 0.2),                '1:Notch0_pos[H]': (0.0, 0.1, 0.1, 0.2, 0.2),
    '1:Notch0_pos[V]': (0.0, 0.1, 0.1, 0.2, 0.2),           '1:Notch1_pos[H]': (0.0, 0.1, 0.1, 0.2, 0.2),
    '1:Notch1_pos[V]': (0.0, 0.1, 0.1, 0.2, 0.2),           '1:Notch2_pos[H]': (0.0, 0.1, 0.1, 0.2, 0.2),
    '1:Notch2_pos[V]': (0.0, 0.1, 0.1, 0.2, 0.2),           '1:Tab height 0': (3.91, 0.1, 0.1, 0.2, 0.2),
    '1:Tab height 1': (3.91, 0.1, 0.1, 0.2, 0.2),          '1:Tab height 2': (3.91, 0.1, 0.1, 0.2, 0.2),
}
```

Raw data

Program name	0:Flatness R	0:Flatness L	1:Width edge 0	1:Width corner 0	
CuW_LD_Right_Combined	0.033	0.031	166.987	94.118	
CuW_LD_Right_Combined	0.089	0.108	166.958	94.134	.....
CuW_LD_Right_Combined	0.104	0.125	166.956	94.037	
CuW_LD_Right_Combined	0.052	0.067	166.947	94.041	
		.....			

- 25 entries for grading
- Red grade: Extracted directly from raw output.
- Orange grade: Typically half the value of the red grade, some special cases
- Exclude entries labeled as user, ARC, or tab.()
- Remove test rows, duplicated ones keep the last
- Convert date/time fields to a UTC ISO 8601 format format.

full_graded.csv

Program name	0:Flatness R	0:Flatness L	1:Width edge 0	1:Width corner 0		1:Width edge 0_Grade	1:Width corner 0_Grade	
CuW_LD_Right_Combined	0.033	0.031	166.987	94.118		green	green	
CuW_LD_Right_Combined	0.089	0.108	166.958	94.134	.....	green	green	.....
CuW_LD_Right_Combined	0.104	0.125	166.956	94.037		green	green	
CuW_LD_Right_Combined	0.052	0.067	166.947	94.041		green	green	
		.....						

- 65 columns * # of BPs
- ~ 5kb per 10 BPs

20250401_CuW_LD_Right.json


```
[
 {
  "postqc_date": "2025-04-01T13:22:04.000Z", "workshop_serial": 488, "postqc_user": "chreisners",
  "flatness_r": 0.033, "flatness_l": 0.031, "width_edge0": 166.987,
  "width_corner0": 94.118, "radius0": 83.915, "radius1": 74.833,
  "radius2": 83.496, "radius3": 10.203, "radius4": 83.495,
  "radius5": 74.838, "dist_hole_slot": 65.028, "length_slot": 3.518,
  "tab_height0": 3.91, "tab_height1": 3.896, "tab_height2": 3.895,
  "height_lam0": 1.445, "height_lam1": 1.458, "height_lam2": 1.438,
  "height_lam3": 1.441, "height_lam4": 1.43, "height_lam5": 1.463,
  "height_lam10": 1.437, "height_lam11": 1.448, "height_lam12": 1.433,
  "height_lam13": 1.438, "height_lam14": 1.431, "height_lam15": 1.462,
  "width_edge0_grade": "green", "width_corner0_grade": "green", "radius0_grade": "green",
  "radius1_grade": "green", "radius2_grade": "green", "radius3_grade": "green",
  "radius4_grade": "green", "radius5_grade": "green", "dist_hole_slot_grade": "green",
  "length_slot_grade": "green", "flatness_lam_grade": "green", "tolerance_grade": "orange",
  "notch_size_passed": true, "position_of_notches": true, "diameter_hole_passed": true,
  "width_slot_passed": true, "flatness_grade": "green"
 },
 .....
]
```


- Give ptype specific entries
- Drop unnecessary columns
- ~ 57 entries per BP(depends on geom)
- 15kb per 10 BPs

Check entries, upload

- Pull legal entries from ETP HGCAL DB
- Check if all entries legal. If not, write into upload log, throw to problematic folder
- If all entries legal, upload 20250401_CuW_LD_Right.json along with its(raw_output.csv, full_graded.csv, mask.py)
- Technically could also backup keyence program ~ 60mb per measurement per day.
- Alternatively, checking if 20250401KeyenceIM-8030T.bi3 generated with raw outputs contains the critical info we need.
- DB side, incremental storage of backup files with unique timestamps.