1 Introduction

The application `xia2scan` is designed to scan through a list of provided images and print information about the "quality" of the diffraction on the images. This will end up as an executable `xia2-scan(=> .sh/.bat)` and a python program xia2scan.py.

1.1 Appication

This is designed to be useful for

- Selecting the best images for autoindexing from a sweep.
- Optimizing humidity when a humidifier is available.

by printing information about the strength of diffraction in each image.

2 Uses...

This uses the wrappers for labelit.screen and labelit.stats_distl.

3 Dependencies

This application will depend on having access to a user provided beam centre (by implication then an input argument / command line handler) and also diffraction image name parsing to allow searching of a directory for matching images.

4 Use Cases

4.1 UC 1: Humidifier

Requires:

- The correct beam position (for indexing from single images.)
- The images.

Provides:

- The statistics for each image in the list.

4.2 UC 2: Indexing Image Selection

This will scan through all of the images in a sweep and select, based on the spot separation and so on, the best pair for autoindexing.

5 Implementation

5.1 UC 1: Humidifier

This will use the following classes:

- Schema.Sweep.SweepFactory - to generate a list of sweeps from the images provided.
- Handlers.CommandLine - to hold the beam centre information from a previous indexing run.

To achieve reasonable speed, the labelit run will not optimise the beam centre (this is unreliable from a single image anyway) and also will run the labelit before distl, and use labelit.stats_distl to get the results afterwards.

This should do all of the running, then collate the results and print a summary. The following arc should be followed:

- [determine accurate beam centre]
- Digest matching frames to list of sweeps.
- For each sweep, run labelit, get an idea of the unit cell volume, run labelit.stats_distl to get the intensity summary, save.
- Print summary by image number.

5.1.1 Test Data

Rajan provided me with a couple of test data sets - use these. Oh bugger - the images are stills... Add a special case in the sweep factory to return these as separate frames? Or just plough ahead??

5.1.2 Results

This produces the following results:

```
.....................................................
  1   1512   1080   1.82   1.95   0.13    426182
  2   1465   1076   1.79   1.85   0.13    422625
  3   1498   1088   1.83   1.84   0.30    422697
  4   1307    899   1.86   1.67   0.13    425974
  5   1519   1104   1.81   1.84   0.10    423063
  6   1480   1083   1.81   1.88   0.13    422251
  7   1536   1107   1.82   1.87   0.13    422419
  8   1480   1140   1.81   1.80   0.25    422219
  9   1519   1178   1.80   1.83   0.20    424185
 10   1483   1117   1.82   1.81   0.20    424605
 11   1471   1098   1.83   1.83   0.20    423895
```

12	1459	1080	1.83	1.84	0.35	423090
13	1398	1041	1.83	1.80	0.07	424593
14	1515	1135	1.85	1.85	0.13	421315
15	1412	1067	1.86	1.82	0.13	421680
16	1408	1055	1.76	1.84	0.13	422815
17	1448	1133	1.78	1.77	0.13	425121
18	1472	1059	1.81	1.94	0.30	419121
19	1428	1080	1.84	1.81	0.20	423461
20	1362	1021	1.86	1.79	0.10	423779
21	1340	935	1.84	1.89	0.13	420958
22	1277	889	1.82	1.88	0.07	422502
23	1438	1104	1.84	1.83	0.35	421221
24	1379	959	1.84	1.87	0.13	421714
25	1280	946	1.83	1.83	0.10	425501
26	1388	1037	1.84	1.82	0.13	422932
27	1444	1116	1.86	1.81	0.15	420991
28	1352	1024	1.77	1.83	0.15	421341
29	1377	1066	1.77	1.86	0.15	423309
30	1430	1113	1.86	1.80	0.13	421519
31	1371	1039	1.79	1.83	0.10	422713
32	1252	938	1.83	1.88	0.15	422474
33	1343	1007	1.84	1.88	0.45	426324
34	1346	944	1.82	1.93	0.13	423602
35	1355	1037	1.82	1.87	0.25	420853
36	1182	851	1.81	1.94	0.10	425738
37	1219	917	1.83	1.94	0.15	422869
38	1231	882	1.79	1.95	0.13	423181
39	1273	922	1.84	1.83	0.15	424329
40	1302	954	1.86	1.87	0.20	424986
41	1358	1036	1.82	1.88	0.20	424799
42	1206	844	1.81	1.94	0.10	424697
43	1291	983	1.86	1.87	0.25	425787
44	1205	930	1.80	1.86	0.15	428759
45	1151	910	1.88	1.86	0.13	423216
46	1132	864	1.84	1.87	0.30	423688
47	1183	911	1.82	1.85	0.13	424232
48	1161	902	1.81	1.84	0.25	423901
49	1152	850	1.84	1.87	0.15	422314
50	1127	843	1.84	1.93	0.13	428840
51	1060	812	1.80	1.87	0.07	423521
52	1205	891	1.89	1.96	0.15	423589
53	1269	983	1.84	1.88	0.13	420966
54	1257	961	1.87	1.88	0.25	419352
55	1254	989	1.85	1.87	0.15	428291
56	1202	918	1.80	1.88	0.30	435571
57	1167	914	1.79	1.87	0.13	423883
58	1224	998	1.84	1.79	0.20	424434
59	1056	811	1.82	1.87	0.25	418296
60	1159	852	1.86	1.87	0.10	419464

The dots are printed on per image, and the columns are image number, total number of spots, good spots, resolutions (2), mosaic and unit cell volume in Å cubed.