The AstroWideImageMapper (AWIM) project aims to enable users to label "wide-angle" images of any format with with pixel-by-pixel directional astronomical coordinate data, i.e. directional azimuth and altitude.

In a world of seemingly-infinite technology, some notes to avoid “re-inventing the wheel” and to enable maximum future flexibility:

1. JPEG, PNG, and RAW are sufficient to cover every type of human-viewable 2-D image one can imagine, so let’s stick with them. JPEG covers an extreme range of resolutions with sufficient compression for reasonable file sizes. PNG includes transparency at the expense of larger file size. RAW covers uncompressed high-detail ignoring file size.
2. See list of related / similar software at:

<https://timev3technology.com/astrowideimagemapper-and-related-similar-software/>

1. Users should be able to calibrate their own camera with readily-available equipment. The goal is to demystify the idea of angular direction of photographs.
2. Location (on Earth) and time data are already covered extensively by EXIF, both within the GPS section and outside of it.
3. Altitude: directional altitude of a photograph is easily obtainable digitally by gravity sensors that know a camera’s direction to fraction-of-a-degree accuracy. However! The data is not included in EXIF and even top-end professional cameras do not show this data to the user.
   1. Professional cameras (to my knowledge) show only “level or not,” they do not give the user the actual number.
   2. There exist various \*free\* “angle finder” phone apps that work great because of built-in sensors in modern smart phones.
   3. Angle finders are available for carpenters for about $30 at Home Depot, Amazon, etc. (annoyingly, almost all contain an extremely strong magnet designed to allow the user to stick the device to whatever metal is to be leveled – I was nervous about the magnet near my camera so I removed the magnet).
4. Azimuth is not easy digitally. The primary physical phenomenon that would tell an electronic device its azimuth orientation is Earth’s magnetic field. Unlike gravity and besides magnetic variation, which is easily-calculable, Earth’s magnetic field is weak and very distorted. For example, using phone apps and compasses alike in a relatively rural location, I was recording azimuth values that were +/- 30°. Speculation: difficulty in azimuth is probably the sole reason directional data is not already included in EXIF.
   1. Question: can GPS signals be used by a receiver with directional antennae to determine its azimuth orientation? I have never heard this question asked.
5. The data should be text. While there may be some efficiency advantage to some other data format, modern computers can quickly convert text into whatever format it prefers and HUMANS CAN READ TEXT. (some EXIF data is specifically not text, and it is very annoying to work with). Being that EXIF is already so widely distributed, AWIM data should “piggy-back” onto or be added directly to existing EXIF data.

Finally, an example of AWIM data text, generated by the AstroWideImageMapper:

Center Pixel: 959.5, 539.5

Center AzAlt: 180.0, 50.0

Pixel Models: ,xang,yang,xang^2,xang yang,yang^2,xang^3,xang^2 yang,xang yang^2,yang^3

x\_px\_predict,19.292452739173907,-0.24956886464055963,-0.11680583951024957,0.002134526371052793,0.01594867120694733,0.004105889788502424,8.422780268702468e-05,-9.285313613953683e-05,-0.000213317513574407

y\_px\_predict,0.044698939975893144,17.51717668924327,-0.0027842723023968575,-0.03962252166133535,0.03855873423870494,2.412096065654623e-05,-6.933121515999119e-05,0.0013393358474758088,0.0006276904237535063

Pixel Map Type: 3d\_degree\_poly\_fit\_abs\_from\_center

x,y Angle Models: ,x\_px,y\_px,x\_px^2,x\_px y\_px,y\_px^2,x\_px^3,x\_px^2 y\_px,x\_px y\_px^2,y\_px^3

xang\_predict,0.05580199939400039,-9.664180644328605e-05,-9.344329268187512e-07,2.4839749430340396e-07,-2.3162935561304018e-07,-8.582985193235125e-10,-7.274944647375925e-11,-2.1275620147643967e-11,5.917972194688185e-11

yang\_predict,-0.00020714796873958584,0.057816858705502815,2.507352690413978e-07,1.8165108785404635e-06,-3.6687089945097524e-06,-5.0456156180239686e-11,-1.6090319181205626e-11,-9.882743508640192e-10,2.5252182343869084e-10

Pixel Borders: [-959.5 539.5], [ 0. 539.5], [959.5 539.5], [-959.5 0. ], [0. 0.], [959.5 0. ], [-959.5 -539.5], [ 0. -539.5], [ 959.5 -539.5]

x,y Angle Borders: [-43.26839682 28.49500958], [-0.17206967 28.24250612], [43.26839682 28.49500958], [-43.44953595 0.0873467 ], [0. 0.], [43.44953595 0.0873467 ], [-43.26839682 -28.49500958], [ -0.17206967 -28.24250612], [ 43.26839682 -28.49500958]

Degrees per Hundred Pixels at Center: : 5.579907090928261, 5.780539645592841

"Wide-angle" in this case specifically means NOT a telescope where the image can be assumed to be flat but rather any camera like a point-and-shoot, smartphone, professional mirrorless, etc. where the direction of all the pixels is not linear. In particular, ultra-wide-angle lenses favored for astrophotography are particularly curved near the edges.

In order to map the pixels of a wide-angle image, the properties of the camera and lens system must be known in more detail along with sufficiently specific settings data.

There must be a standard way to define and manipulate such data. It will be useful to enable the user to gather tens or hundreds of pixel direction data from a camera / lens / settings system one time and after gathering a manageable number of data points let the software fill in an estimation for the rest of the millions of pixels

 gathered by modern commercially-available cameras - or some useful fraction thereof.

 The extreme camera example would be, for example, a fish-eye lens where image distortion is purposeful and extreme, obviously not flat, the opposite of a telescope.

 Some assumptions (that I believe are valid) make the challenge possible:

 1. Any camera, where the lens and settings are specifically defined, points its pixel sensors in specific non-changing direction relative to the other pixels.

 2. If the location (on Earth or the surface of any cellestial object) and orientation of the camera at the time of the image collection is known,

    each pixel can be assigned an altitude/azimuth coordinate relative to the cellestial object that does not change over time.

 The following are required:

 A new container class, CameraAim, to define the properties of a particular camera / lens / settings system, especially the direction of each pixel or manageable fraction

 of the pixels relative to the a reference pixel.

 CameraAim is specific to a camera with a specific lens and specific settings.

 A new function(?) / method(?) within CameraAim to combine the information from

 1. the CameraAim class

 2. astroplan.Observer class

 3. orientation data

 4. the capture moment of an image

 and return a class "Snapshot" that for a particular image that includes

 1. astropy.coordinates.builtin\_frames.AltAz object with NumPy arrays for altitude and azimuth that correspond to the altitude and azimuth of

 each pixel in the image - or some manageable and useful fraction of the pixels.

 2. capture moment of the image

 3. name of camera.

 4. etc.

 A new function within CameraAim that efficiently saves CameraAim data to disk for future use.

 A new function within CameraAim that

 Humans are good at orienting themselves with respect to 2-dimensional spaces we can see and travel within. The cosmos being largely not visible and 3-dimensional

 and being that we only travel in a repeating relatively tiny ellipse throughout our lives and being that we constantly rotate and we constantly orbit around

 the brightest light within said space, astonomers are accustomed to maintaining a mental model of the cosmos and lay persons are accustomed to being essentially lost within it.

 Software is often used to cover up challenging questions with instant answers and enable humans to un-know things.

 Software can also be used to shine light on questions and enable humans to challenge themselves to ask ask questions previously thought un-knowable by shining light on

 questions rather than the answers.

 Coding data standardization notes:

 - Any coordinate values are stored as a list of two floats with the horizontal value first, i.e. [azimuth, altitude], [x pixel, y pixel], [latitude, longitude]

 - for lists of coordinates, store as a 2-dim, two-column Numpy array OR a 3-dim Numpy array such that the 0-layer is horizontal, 1-layer is vertical

 - All time information is stored as  "aware" datetime object where date is Gregorian NS, time is UTC. datetime tzinfo source is datetime.timezone.UTC