For correcting angle or perspective of an image there are several algorithms and techniques we can use, depending on your specific needs

* Homography based corrections: these type of corrections is based on homographic images for fixing mostly documented related scanning

1. RANSAC (Random Sample Consensus): used to estimate the homography matrix by repeatedly selecting subsets of points to find a transformation that is best aligns the images.
2. Direct Linear Transformation (DLT): compute the homography matrix directly from point match without a lot estimation.

* Affine Transformation: this type of corrections is for rotating, skewing, translating, and scaling and their affine parameters calculation

1. Least Squares Estimation: used to find the best affine transformation parameters that aligns points in the image.

* Perspective transformation: this type of corrections is for perspective distortions

1. Projective transformation: adjust the perspective of an image using a 3x3 matrix to map points from one plane to another.

* Rectification Using Vanishing Points: this type of corrections is for images of scene with parallel lines that has perspective distortion

1. Epipolar Geometry: in stereo vision, rectification aligns images so that matching points lie on the same horizontal line.
2. Plane Based Rectification: maps points onto a specific plane to correct perspective distortions.

* Deep Learning Approaches: recent advancements have led to neural networks and deep learning models that can learn to correct perspective or angels in images. These methods often require training on large datasets but can be very effective.

1. Convolutional Neural Networks (CNNs): use CNNs to learn perspective corrected from a dataset of images with known distortions and corrections.
2. Generative Adversarial Networks (GANs): employ GANs to generate corrected images based on learned patterns.

* Feature Based Methods:

1. SIFT (Scale Invariant Feature Transform) and SURF (Speed UP Robust Feature): detect and match features across images to estimate transformations.
2. ORB (Oriented Fast and Rotated Brief): provides sufficient feature detection and description for perspective correction.

In another point of view, we can split them like below:

1. Homography-based Correction:
2. Direct Linear Transformation (DLT): A method for computing a homography matrix that maps points from one perspective to another.
3. RANSAC (Random Sample Consensus): Often used in conjunction with DLT to robustly estimate homographies by ignoring outliers.
4. Camera Calibration Techniques:
5. **Zhang’s Method:** A popular technique for estimating intrinsic and extrinsic parameters of a camera, which helps in perspective correction.
6. Warping Techniques:
7. **Affine Transformation:** Adjusts the image to correct for distortions by applying affine transformations (translation, scaling, rotation, and shearing).
8. **Projective Transformation:** Corrects perspective distortions by applying a projective transformation, which can handle more complex distortions than affine transformations.
9. Rectification Techniques:

a. **Epipolar Geometry:** Used in stereo vision to rectify images such that corresponding points lie on the same horizontal line.

b. **Image Rectification:** Corrects images to make them appear as though they were taken from the same viewpoint or plane.

1. Deep Learning Approaches:
   1. **Convolutional Neural Networks (CNNs):** Used for learning perspective correction tasks from large datasets.
   2. **Generative Adversarial Networks (GANs):** Can be trained to correct perspective distortions by generating corrected images that align with a target perspective.
2. Vanishing Point Detection:
   1. **Hough Transform:** Used for detecting vanishing points in images to assist in perspective correction by aligning the image to these points.
3. Transform-based Methods:
   1. **Fourier Transform:** Applied to correct distortions in the frequency domain, though less common for perspective correction.
4. Optimization-based Methods:
   1. **Non-linear Optimization:** Used to fine-tune transformations by minimizing the difference between the corrected and target images.

When categorizing perspective correction algorithms by their "lightness" or computational complexity, it's useful to consider both their theoretical complexity and practical implementation. Here’s a general classification from relatively lighter to more complex algorithms:

Affine Transformation: more for skewing, rotation and so on

* Light
* Applies basic transformations like translation, scaling, rotation, and shearing. Simple to compute and implement.

Direct Linear Transformation (DLT): best for this project

* Light to Moderate
* Calculates a homography matrix using a set of point correspondences. Computationally manageable but requires precise point matching.

Vanishing Point Detection: not for perspective correction, actully for image

stacking

* Light to Moderate
* Techniques like Hough Transform are used to detect vanishing points, which is generally less computationally intensive but can vary based on image complexity.

Projective Transformation: second best for this project

* Moderate
* Handles more complex perspective distortions compared to affine transformations. Requires solving a system of equations which adds moderate complexity.

Camera Calibration Techniques: require a lot images

* Moderate
* Estimates camera parameters for perspective correction. The process involves solving nonlinear optimization problems which are computationally more intensive.

Image Rectification: require numpy package for image stacking

* Moderate
* Aligns images to a common plane, which can be more complex than simple transformations due to the need to handle image pairs and calibration.

RANSAC (Random Sample Consensus): third best for this project

* Moderate to Heavy
* Often used with DLT to robustly estimate homographies by iteratively finding consensus. The iterative nature increases computational demands.

Deep Learning Approaches (e.g., CNNs, GANs): require model for predictions and image generations also need a lots of image for training and testing

* Heavy
* Involves training neural networks on large datasets for perspective correction. The training process is computationally expensive and requires significant resources.

Non-linear Optimization Techniques: need a lots of computation and also it is for model optimization

* Heavy
* Used for fine-tuning transformations by minimizing errors, which involves solving complex optimization problems that can be computationally intensive.