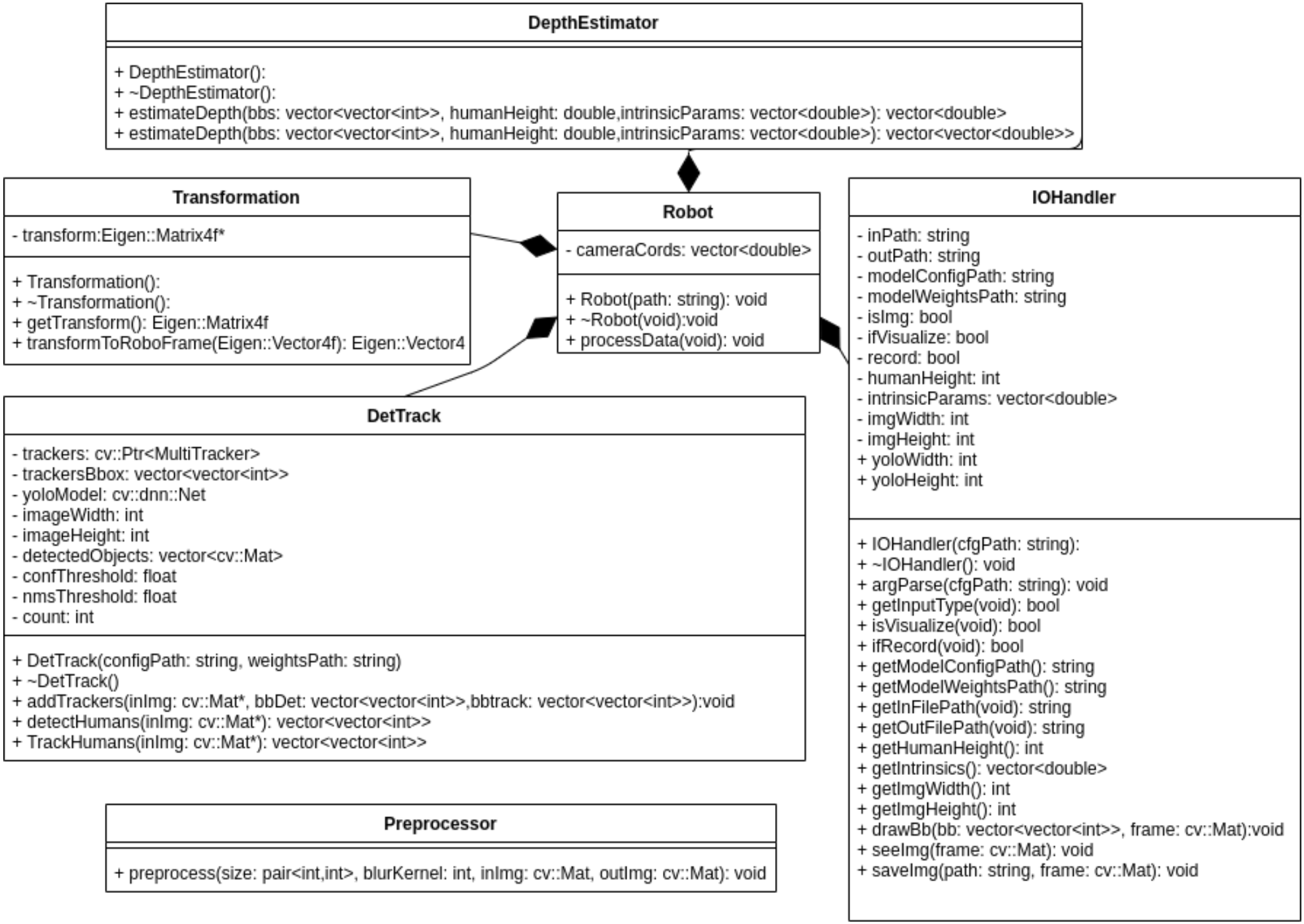
DepthEstimator
+ DepthEstimator():
+ ~DepthEstimator():
+ estimateDepth(bbs: vector<vector<int>>, humanHeight: double,intrinsicParams: vector<double>): vector<double>
+ estimateDepth(bbs: vector<vector<int>>, humanHeight: double,intrinsicParams: vector<double>): vector<vector<double>>
Transformation
- transform:Eigen::Matrix4f*
+ Transformation():
+ ~Transformation():
+ getTransform(): Eigen::Matrix4f
+ transformToRoboFrame(Eigen::Vector4f): Eigen::Vector4
Robot
- cameraCords: vector<double>
+ Robot(path: string): void
+ ~Robot(void):void
+ processData(void): void
IOHandler
- inPath: string
- outPath: string
- modelConfigPath: string
- modelWeightsPath: string
- isImg: bool
- ifVisualize: bool
- record: bool
- humanHeight: int
- intrinsicParams: vector<double>
- imgWidth: int
- imgHeight: int
+ yoloWidth: int
+ yoloHeight: int
+ IOHandler(cfgPath: string):
+ ~IOHandler(): void
+ argParse(cfgPath: string): void
+ getInputType(void): bool
+ isVisualize(void): bool
+ ifRecord(void): bool
+ getModelConfigPath(): string
+ getModelWeightsPath(): string
+ getInFilePath(void): string
+ getOutFilePath(void): string
+ getHumanHeight(): int
+ getIntrinsics(): vector<double>
+ getImgWidth(): int
+ getImgHeight(): int
+ drawBb(bb: vector<vector<int>>, frame: cv::Mat):void
+ seeImg(frame: cv::Mat): void
+ saveImg(path: string, frame: cv::Mat): void
DetTrack
- trackers: cv::Ptr<MultiTracker>
- trackersBbox: vector<vector<int>>
- yoloModel: cv::dnn::Net
- imageWidth: int
- imageHeight: int
- detectedObjects: vector<cv::Mat>
- confThreshold: float
- nmsThreshold: float
- count: int
+ DetTrack(configPath: string, weightsPath: string)
+ ~DetTrack()
+ addTrackers(inImg: cv::Mat*, bbDet: vector<vector<int>>,bbtrack: vector<vector<int>>):void
+ detectHumans(inImg: cv::Mat*): vector<vector<int>>
+ TrackHumans(inImg: cv::Mat*): vector<vector<int>>
Preprocessor
+ preprocess(size: pair<int,int>, blurKernel: int, inImg: cv::Mat, outImg: cv::Mat): void