DESIGN AND IMPLEMENTATION OF A MONOCULAR MEASUREMENT SYSTEM ON ANDROID USING DYNAMIC CAMERA INTRINSICS

Trung Duc Bui
MSE28HN
FPT School of Business
Technology
Hanoi, Vietnam
trungbuiducbuiduc@gmail.com

December 17, 2025

ABSTRACT

Estimating real-world object dimensions from smartphone images is a fundamental problem in computer vision, with applications in e-commerce, logistics, and interior design. Commercial solutions typically rely on augmented reality (AR) frameworks such as ARCore or ARKit, which provide metric scale through SLAM-based 3D reconstruction. However, these frameworks act as "black boxes" that obscure the underlying geometric principles, impose operational constraints, and—critically—are not available on all devices, as ARCore certification covers only a subset of Android smartphones. This paper presents the design and implementation of a *traditional, AR-free measurement pipeline* for Android that operates independently of proprietary AR SDKs, enabling metric measurement on any device with Camera2 API support. The proposed system implements a *Dynamic Intrinsics* mechanism, recalculating the camera matrix K per-frame based on `SCALER_CROP_REGION` metadata to maintain accuracy across continuous digital zoom levels. By fusing optical data with IMU-derived orientation, the system supports classical measurement models including ground-plane homography, planar object rectification, and single-view metrology. The application features a dual-mode design: a simplified *User Mode* for practical measurement tasks, and a dedicated *Researcher Mode* that exposes internal parameters and enables experimental analysis. Systematic experiments on a Xiaomi Poco X6 Pro demonstrate that classical computer vision techniques, when properly controlled for intrinsic parameter variations, can achieve practical accuracy while offering full transparency, interpretability, and broad device compatibility.

Keywords Single-View Metrology · Dynamic Camera Intrinsics · Android Camera2 API · Digital Zoom Compensation · Pinhole Camera Model · Ground-Plane Homography · Mobile Measurement

1 Introduction

Estimating real-world object dimensions from images captured by consumer smartphones is a fundamental problem in computer vision, with practical applications spanning e-commerce (product sizing), logistics (package dimensioning), interior design (room measurement), and augmented reality. Over the past decade, mobile devices have witnessed remarkable advances in optical hardware, equipping modern smartphones with high-resolution sensors, optical image stabilization (OIS), and low-level hardware access via APIs such as Android's Camera2. While this evolution positions smartphones as potential precision instruments for metrology, realizing a mathematically transparent measurement tool remains a significant challenge.

Modern smartphones possess powerful cameras and sensors capable of metric measurement, yet most practical solutions rely on proprietary frameworks that obscure the underlying geometric principles. These frameworks, while convenient,

act as "black boxes" that hide implementation details, impose operational constraints (requiring camera motion for initialization, texture-rich environments), and are often unavailable on many devices. For researchers, educators, and learners seeking to understand, validate, or extend measurement algorithms, such opacity and limited availability represent significant barriers.

A particularly underexplored challenge within this domain is the *dynamic nature of camera intrinsics during digital zoom operations*. Unlike optical zoom which physically adjusts lens elements, digital zoom fundamentally alters the effective camera matrix through sensor cropping and resampling. The intrinsic parameters—focal length and principal point—change with each zoom level, yet standard measurement applications often assume a fixed camera matrix, leading to systematic errors that compound at higher magnification. This problem is exacerbated by the fact that few existing systems leverage the rich per-frame metadata available through the Camera2 API, particularly `SCALER_CROP_REGION`.

This paper addresses these challenges by designing a *fully transparent measurement pipeline* that applies classical computer vision techniques to the smartphone measurement problem. Drawing on foundational methods in camera calibration [4], projective geometry [5], and single-view metrology [6], the proposed system demonstrates that transparent, interpretable approaches can achieve practical accuracy while enabling deep understanding and experimentation.

The system implements several key technical contributions. First, a *Dynamic Intrinsics* mechanism continuously recalculates the camera matrix K based on per-frame `SCALER_CROP_REGION` metadata from the Android Camera2 API. This addresses a critical limitation of fixed-calibration approaches: when users apply digital zoom, the effective crop region changes, and the intrinsic parameters must be recomputed to maintain accuracy. Second, the pipeline integrates multiple classical measurement modules: (1) ground-plane homography for objects on flat surfaces, leveraging known camera height and IMU orientation; (2) perspective rectification for planar objects using four-point homography; and (3) single-view metrology using vanishing point detection for vertical structures. Third, the image processing pipeline applies classical techniques including Canny edge detection, Harris corner refinement, subpixel localization, and RANSAC outlier rejection to improve robustness.

Uniquely, the application features a dual-mode design: a simplified *User Mode* for end-users requiring quick, practical measurements, and a *Researcher Mode* that exposes all internal parameters, supports toggling of processing steps (lens undistortion, edge-based point snapping, perspective rectification), displays debugging overlays (principal point, grid, detected edges, IMU orientation), and enables export of detailed measurement logs (JSON/CSV format) for offline analysis. This design serves both practical measurement needs and educational/research objectives.

The proposed pipeline is validated through systematic experiments on a Xiaomi Poco X6 Pro, a mid-range Android device with full Camera2 API support. Experimental results demonstrate that the Dynamic Intrinsics mechanism is essential for maintaining accuracy across zoom levels, and that classical geometric techniques—when properly implemented with careful attention to calibration, distortion correction, and subpixel refinement—can achieve practical measurement accuracy while providing full transparency, interpretability, and reproducibility.

The main contributions of this work are:

1. A **Dynamic Intrinsics mechanism** that recalculates the camera matrix K_{out} per-frame based on `SCALER_CROP_REGION` metadata, maintaining measurement accuracy across continuous zoom levels where fixed-calibration approaches fail.

2. Integration of **multiple classical measurement modules**: ground-plane homography with IMU fusion, perspective rectification for planar objects, and single-view metrology with vanishing points.

3. A **comprehensive image processing pipeline** applying edge detection (Canny), corner refinement (Harris, subpixel), contour analysis, and RANSAC outlier rejection to improve measurement robustness.

4. A **dual-mode application design** with User Mode for practical measurements and Researcher Mode exposing all parameters, processing toggles, debug overlays, and detailed logging for educational and research purposes.

5. **Systematic experimental validation** demonstrating the importance of Dynamic Intrinsics across zoom levels and quantifying the impact of individual processing options (undistortion, edge snapping).

The remainder of this paper is organized as follows. Section 2 reviews related work on camera calibration, single-view metrology, and existing measurement approaches. Section 3 describes the proposed system architecture and measurement modules. Section 4 presents the experimental setup and methodology. Section 5 reports results and discussion. Section **??** concludes the paper, and Section 7 outlines directions for future work.

2 Literature Review

This section establishes the theoretical foundations of the proposed system, covering the geometric modeling of cameras, the specific challenges of digital image formation on mobile devices, and existing techniques for extracting metric dimensions from 2D images.

2.1 Pinhole Camera Model and Calibration

2.1.1 The Pinhole Model

The pinhole camera model serves as the fundamental geometric abstraction for image formation in computer vision [5]. This idealized model assumes that all light rays pass through a single point (the optical center) before striking the image plane. Under this model, a 3D world point $\mathbf{X}_w = (X, Y, Z)^T$ is projected onto the 2D image plane at pixel coordinates (u, v) through a linear transformation in homogeneous coordinates:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R \mid t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{1}$$

where s is an arbitrary scale factor arising from the use of homogeneous coordinates. The 3×4 projection matrix $P = K[R \mid t]$ decomposes into two components: the *extrinsic parameters* $[R \mid t]$ describing the camera's pose (rotation $R \in SO(3)$ and translation $t \in \mathbb{R}^3$) in world coordinates, and the *intrinsic matrix* K encoding the internal camera geometry:

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Here, f_x and f_y represent the focal length expressed in pixel units along the horizontal and vertical axes respectively, (c_x, c_y) denotes the principal point (the intersection of the optical axis with the image plane), and s is the skew coefficient (typically zero for modern sensors with rectangular pixels).

2.1.2 Lens Distortion

Real camera lenses deviate from the ideal pinhole model by introducing geometric distortions. The two primary types are:

Radial distortion causes straight lines to appear curved, particularly near the image periphery. It is modeled as a polynomial function of the distance r from the principal point:

$$\begin{aligned} x_{distorted} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{distorted} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{3}$$

where (x, y) are normalized image coordinates and k_1, k_2, k_3 are the radial distortion coefficients. Positive coefficients produce barrel distortion (lines curve outward), while negative coefficients produce pincushion distortion (lines curve inward).

Tangential distortion arises from imperfect alignment between the lens elements and the image sensor, causing asymmetric displacement:

$$\begin{aligned} x_{distorted} &= x + [2p_1 xy + p_2(r^2 + 2x^2)] \\ y_{distorted} &= y + [p_1(r^2 + 2y^2) + 2p_2 xy] \end{aligned} \tag{4}$$

where p_1, p_2 are the tangential distortion coefficients.

For accurate metric measurement, these distortions must be corrected through an *undistortion* process that maps observed pixel coordinates to their ideal positions.

2.1.3 Camera Calibration Techniques

Camera calibration is the process of estimating the intrinsic parameters K and distortion coefficients from images of known patterns. Zhang's flexible calibration method [4] revolutionized this process by requiring only a planar calibration target (typically a checkerboard) captured at multiple orientations.

The procedure involves:

1. **Pattern detection**: Locating grid corners in each calibration image using algorithms such as `findChessboardCorners`.
2. **Subpixel refinement**: Improving corner localization accuracy to 0.1 pixels or better using gradient-based methods (`cornerSubPix`).
3. **Homography estimation**: Computing the 3×3 homography H_i mapping pattern coordinates to image coordinates for each view.
4. **Intrinsic extraction**: Exploiting constraints on the absolute conic to solve for K from multiple homographies.
5. **Extrinsic recovery**: Computing rotation and translation for each view given K.
6. **Nonlinear refinement**: Minimizing the total reprojection error over all views using Levenberg-Marquardt optimization.

The reprojection error—the RMS distance between observed corners and their predicted positions—serves as a quality metric; values below 0.5 pixels indicate excellent calibration.

2.1.4 Advanced Calibration Targets: ChArUco

While traditional checkerboards are widely used, they suffer from a significant limitation: they require the entire board to be visible to establish the board coordinate system. This constrains the user to keep the full board in the camera's field of view, making it difficult to collect corner data at the extreme edges of the image where lens distortion is most severe.

To address this, ChArUco boards [12] were developed as a hybrid of checkerboards and ArUco fiducial markers (see Figure 1a). A ChArUco board places unique ArUco markers inside the white squares of a checkerboard. This design offers two critical advantages:

1. **Partial Occlusion Robustness**: Unlike standard checkerboards, ChArUco boards do not require the entire pattern to be visible. The system can uniquely identify any visible subset of the board using the ArUco IDs, allowing for calibration even when corners are occluded or cut off by the frame edge.
2. **Improved Corner Coverage**: Because partial visibility is supported, users can move the board close to the camera boundaries, capturing data points in the high-distortion peripheral regions that are essential for accurate distortion modeling.
3. **High Precision**: The checkerboard structure retains the high-precision corner detection properties of grid patterns (using saddle-point refinement), while the ArUco markers provide robust identification.

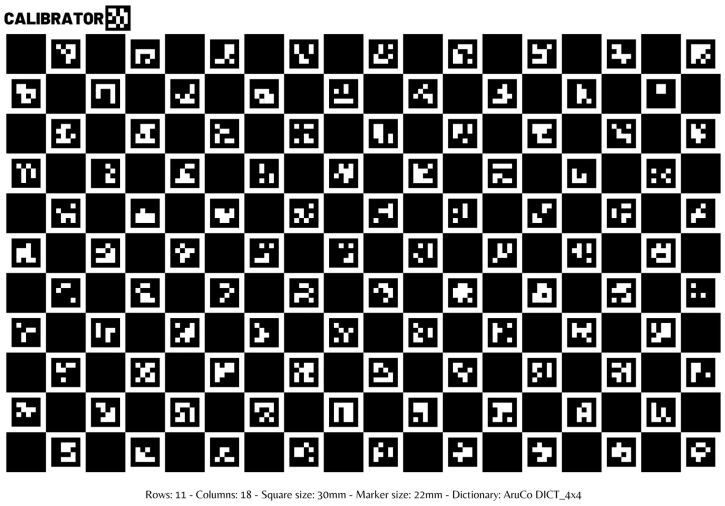
This work employs ChArUco calibration to ensure robust estimation of distortion coefficients, particularly in the peripheral regions of the smartphone lens.

2.2 Single-View Metrology

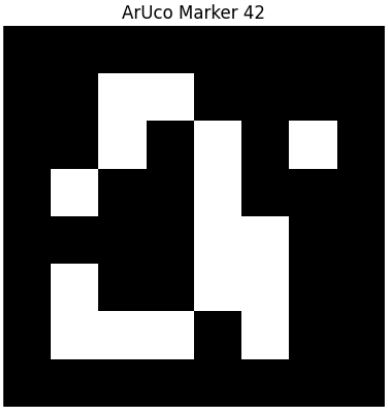
Single-view metrology addresses the recovery of metric information from a single 2D image using projective geometry constraints [6]. This is fundamentally an ill-posed problem since depth information is lost during projection; however, additional constraints—such as known geometry, scene planarity, or reference dimensions—can make metric reconstruction tractable.

2.2.1 Homography and Ground-Plane Measurement

When measuring objects that lie on a known plane (e.g., the ground or a table surface), the image-to-world mapping simplifies considerably. For a plane at $Z = 0$ in world coordinates, the projection equation reduces to a 3×3 homography H:



(a) ChArUco board pattern



(b) ArUco marker

Figure 1: Comparison of calibration patterns: (a) ChArUco board combining checkerboard corners with ArUco markers, and (b) standard ArUco marker grid.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}, \quad H = K[r_1 \mid r_2 \mid t] \tag{5}$$

where r_1, r_2 are the first two columns of the rotation matrix R and t is the translation. Given knowledge of the camera's intrinsic matrix K, height h above the ground plane, and orientation from an inertial measurement unit (IMU), the homography H can be computed analytically. The inverse homography H^{-1} then maps any image point (u, v) to ground coordinates (X, Y), enabling direct distance computation:

$$d = \sqrt{(X_A - X_B)^2 + (Y_A - Y_B)^2} \tag{6}$$

This approach is particularly effective for measuring horizontal distances on floors, tables, or other flat surfaces when the camera pose is known.

2.2.2 Planar Object Rectification

For planar objects not aligned with the ground (e.g., a wall-mounted poster, a box face, or a sheet of paper), perspective rectification enables measurement. Given four corner points of a rectangular object in the image, a homography H_{rect} can be computed to warp the object to a fronto-parallel (orthogonal) view:

$$H_{rect} : (u_i, v_i) \rightarrow (x'_i, y'_i), \quad i = 1, 2, 3, 4 \tag{7}$$

In the rectified image, the object appears as a rectangle without perspective distortion. If the approximate distance Z from the camera to the object plane is known (e.g., from autofocus metadata or user input), the pixel-to-metric scale can be derived from the pinhole model:

$$L_{real} = \frac{L_{pixels} \cdot Z}{f} \tag{8}$$

where f is the focal length and L_{pixels} is the measured length in the rectified image.

2.2.3 Vanishing Points and Cross-Ratio

For scenes with strong perspective effects—such as tall buildings, corridors, or streets—vanishing point analysis provides powerful geometric constraints [7]. Parallel lines in 3D space converge to a vanishing point (VP) in the image; three mutually orthogonal vanishing points (corresponding to the X, Y, and Z world axes) fully constrain the camera's orientation.

The cross-ratio, a projective invariant preserved under perspective projection, enables height estimation from a single image. Given a vertical structure with its base at ground level, if the camera height h_{cam} is known and the horizon line (connecting horizontal vanishing points) is identified, the ratio of image distances can be used to compute the unknown object height h_{obj}:

$$h_{obj} = h_{cam} \cdot \frac{d(base, top)}{d(base, horizon)} \tag{9}$$

where $d(\cdot, \cdot)$ denotes the signed distance along the vertical image direction.

2.3 Why Not Multi-View Triangulation?

Multi-view triangulation is a well-established technique for recovering 3D structure from multiple images, capable of obtaining metric depth without requiring explicit scale assumptions such as known distances or camera height. Given corresponding points across two or more views, the 3D position can be computed by intersecting the back-projected rays from each camera.

However, triangulation critically depends on accurate knowledge of the *camera extrinsics*—specifically, the relative pose (rotation R and translation t) between frames, commonly referred to as the *baseline*. The accuracy of depth recovery is directly proportional to baseline accuracy; even small errors in the estimated translation lead to large errors in triangulated depth, particularly for distant objects.

In systems employing visual SLAM (e.g., ARCore, ORB-SLAM), the baseline is continuously estimated through feature tracking, epipolar geometry constraints, and bundle adjustment optimization. These frameworks provide robust, drift-corrected pose estimation suitable for metric triangulation. However, in the absence of such frameworks, the only available motion source is the device's integrated inertial measurement unit (IMU).

2.3.1 IMU-Based Baseline Estimation and Its Limitations

Estimating camera translation from IMU data requires double-integrating accelerometer readings:

$$\mathbf{p}(t) = \mathbf{p}_0 + \mathbf{v}_0 t + \int_0^t \int_0^\tau (\mathbf{a}(s) - \mathbf{g})\, ds\, d\tau \tag{10}$$

where $\mathbf{a}(t)$ is the measured acceleration and $\mathbf{g}$ is the gravity vector. This double integration is notoriously sensitive to sensor bias and noise:

- **Accelerometer bias**: Even a small constant bias ϵ accumulates quadratically: position error $\propto \epsilon t^2/2$.
- **White noise**: Random noise integrates to a random walk in velocity and a double random walk in position.
- **Gravity alignment**: Any error in estimating the gravity vector directly corrupts the horizontal acceleration estimate.

For typical smartphone-grade MEMS IMUs, position drift can exceed several centimeters within 1–2 seconds of free motion. This drift magnitude is comparable to or larger than the baseline distances achievable in handheld measurement scenarios, rendering IMU-only baseline estimates unreliable for metric triangulation.

2.3.2 Design Decision: Focus on Single-View Methods

Given these constraints, this work focuses on optimizing *single-view* measurement techniques that impose geometric constraints (ground-plane, planarity, vanishing points) rather than relying on multi-view geometry. This design decision offers several advantages:

1. **No motion requirement**: Measurements can be taken from a single, static camera position—faster and more convenient for users.

2. **No drift accumulation**: Each frame is processed independently; there is no error accumulation across time.
3. **Full transparency**: The geometric assumptions (camera height, planar surface, known reference) are explicit and controllable.
4. **Broader device compatibility**: Works on any device with Camera2 API support, without requiring sophisticated visual-inertial odometry pipelines.

2.4 Smartphone Camera Systems and the Android Camera2 API

Modern smartphones integrate sophisticated camera systems that present both opportunities and challenges for metrology applications. This section examines the hardware characteristics and software interfaces relevant to measurement accuracy.

2.4.1 Smartphone Camera Hardware

Contemporary smartphone cameras feature:

- **High-resolution sensors**: Typical resolutions of 12–200 megapixels, with pixel sizes ranging from 0.7 to 1.8 μm.
- **Multi-camera arrays**: Wide, ultra-wide, and telephoto lenses with different intrinsic parameters.
- **Optical Image Stabilization (OIS)**: Physical lens displacement to compensate for hand shake, which can affect the effective optical axis.
- **Autofocus systems**: Phase-detection or contrast-detection AF providing focus distance metadata.
- **Inertial Measurement Units (IMU)**: Accelerometers and gyroscopes providing device orientation.

Unlike professional cameras with fixed, well-characterized lenses, smartphone cameras exhibit significant variation in intrinsic parameters across devices and may not provide accurate factory calibration.

2.4.2 The Camera2 API and Intrinsic Metadata

The Android Camera2 API [8] provides low-level access to camera hardware, exposing metadata critical for geometric applications. Key fields include:

- `LENS_INTRINSIC_CALIBRATION`: A 5-element array (f_x, f_y, c_x, c_y, s) providing manufacturer-calibrated intrinsic parameters in pixels, defined relative to the sensor's active array coordinate system.
- `SENSOR_INFO_ACTIVE_ARRAY_SIZE`: The dimensions (W_s, H_s) of the sensor's light-sensitive area in pixels.
- `SENSOR_INFO_PHYSICAL_SIZE`: The physical dimensions of the active array in millimeters, enabling conversion between pixel and metric focal lengths.
- `LENS_RADIAL_DISTORTION`: Radial distortion coefficients (when available; many devices report this as unavailable).
- `LENS_POSE_ROTATION` and `LENS_POSE_TRANSLATION`: The pose of the camera relative to the device's IMU coordinate frame.

The availability and accuracy of these fields vary significantly across devices. For example, our target device (Xiaomi Poco X6 Pro) provides `LENS_INTRINSIC_CALIBRATION` but reports `LENS_RADIAL_DISTORTION` as unavailable.

2.4.3 Digital Zoom and the Dynamic Crop Region

A critical challenge for measurement applications is handling digital zoom. Unlike optical zoom, which physically adjusts lens elements and changes the optical focal length, digital zoom operates entirely in the image processing pipeline by:

1. Cropping a subregion of the sensor (the "crop region")
2. Scaling the cropped region to the output resolution through interpolation

The `SCALER_CROP_REGION` field in `CaptureResult` specifies the active crop rectangle (x_0, y_0, w_c, h_c) for each captured frame. This dynamic cropping fundamentally alters the effective intrinsic parameters:

- The **principal point shifts** from (c_x, c_y) to $(c_x - x_0, c_y - y_0)$ in crop coordinates.
- The **effective focal length scales** by the ratio $s_x = W_{out}/w_c$ (horizontal) and $s_y = H_{out}/h_c$ (vertical).

The per-frame intrinsic matrix for the output image becomes:

$$K_{out} = \begin{bmatrix} f_x \cdot s_x & 0 & (c_x - x_0) \cdot s_x \\ 0 & f_y \cdot s_y & (c_y - y_0) \cdot s_y \\ 0 & 0 & 1 \end{bmatrix} \tag{11}$$

Failure to account for these dynamic changes leads to systematic measurement errors that grow with zoom level—a problem largely overlooked by existing measurement applications.

2.5 Existing Measurement Approaches

This section surveys existing approaches to smartphone-based measurement, categorizing them into AR-based and traditional methods.

2.5.1 AR-Based Measurement Systems

Commercial measurement applications such as Google's Measure app (using ARCore) and Apple's Measure app (using ARKit) leverage augmented reality frameworks to perform 3D reconstruction in real-time [9]. These systems employ:

- **Visual-Inertial Odometry (VIO)**: Fusing camera images with IMU data to track device motion with centimeter-level accuracy.
- **Simultaneous Localization and Mapping (SLAM)**: Building a sparse 3D map of the environment while simultaneously localizing the camera within it.
- **Plane Detection**: Identifying horizontal and vertical surfaces using geometric analysis of tracked feature points.
- **Depth Estimation**: On supported devices, utilizing time-of-flight (ToF) sensors or stereo cameras to obtain dense depth maps.

Users place virtual anchors on detected surfaces and the system computes Euclidean distances in metric world coordinates. Reported accuracies of 1–3% for objects within 3 meters are common under favorable conditions [10].

However, AR-based approaches exhibit several fundamental limitations:

1. **Opacity and non-transparency**: The geometric internals are hidden within proprietary frameworks, providing no insight into how measurements are computed. Users cannot inspect, validate, or modify the underlying models.
2. **Operational constraints**: SLAM initialization requires camera motion to establish a baseline; tracking can fail in featureless, textureless, or highly reflective environments. The requirement for motion is impractical for quick, single-shot measurements.
3. **Computational overhead**: Continuous SLAM processing imposes significant battery and thermal constraints.
4. **Device requirements**: ARCore and ARKit are not universally available; older or budget devices may lack certification, limiting accessibility.
5. **Environmental dependencies**: Performance degrades in low-light conditions, with moving objects, or on surfaces lacking sufficient texture.

For educational purposes—where understanding the connection between geometric theory (pinhole model, homography, projective invariants) and practical measurement is paramount—such opacity represents a significant barrier.

2.5.2 Traditional Image-Based Measurement

Alternative approaches have explored purely image-based measurement on smartphones using classical computer vision techniques:

- **Fixed calibration methods**: Performing offline camera calibration with checkerboard patterns and storing the intrinsic matrix for subsequent use. However, these approaches typically ignore the effects of digital zoom.

- **Reference object methods**: Placing a known-size object (credit card, coin, ruler) in the scene to establish pixel-to-metric scale. While simple, this adds friction to the measurement workflow.
- **IMU-assisted ground-plane estimation**: Using device accelerometers and gyroscopes to estimate camera orientation, combined with a user-provided camera height to construct ground-plane homographies [11].
- **Vanishing point methods**: Detecting scene geometry from parallel lines to recover camera orientation and apply cross-ratio based measurement [6].

While these methods are transparent and educational, many exhibit significant limitations in practice:

1. **Static intrinsic assumption**: Most systems assume a fixed camera matrix K and fail to update it when users employ digital zoom, leading to systematic errors.
2. **Neglect of Camera2 metadata**: Few systems leverage the rich per-frame metadata available through Camera2, particularly `SCALER_CROP_REGION`.
3. **Limited processing options**: Users cannot easily compare the effects of different processing choices (undistortion on/off, different measurement models).
4. **Lack of experimental validation**: Rigorous comparison against AR baselines under controlled conditions is rarely provided.

2.6 Research Gap and Contributions

The existing literature reveals two significant gaps that this work addresses:

1. **Lack of transparent, educational measurement systems**: Current AR-based tools act as "black boxes," concealing the connection between geometric theory and practical measurement. For learners and researchers seeking to understand, validate, or modify the underlying models, such opacity is a significant barrier. There is a need for a measurement system that exposes every step of the pipeline—from raw sensor data to final metric output—enabling rigorous educational and experimental use.
2. **Limited handling of dynamic intrinsics under zoom**: Few systems leverage the rich Camera2 metadata—particularly `SCALER_CROP_REGION`—to maintain correct intrinsic parameters across continuous zoom levels. This oversight results in zoom-dependent errors that compound at higher magnification. No prior work has systematically validated the dynamic intrinsics approach against both ground truth and AR baselines.

Table 1 summarizes the key characteristics of existing approaches compared to the proposed system.

Table 1: Comparison of existing measurement approaches with the proposed system.

Approach	Transparency	Dynamic Zoom	No AR Required	Researcher Mode
ARCore/ARKit Apps	Low	N/A (3D)	No	No
Fixed Calibration	High	No	Yes	No
Reference Object	High	Partial	Yes	No
Proposed System	**High**	**Yes**	**Yes**	**Yes**

This work addresses these gaps through the following contributions:

1. A **Dynamic Intrinsics mechanism** that recalculates the camera matrix K_{out} per-frame based on `SCALER_CROP_REGION` metadata, maintaining measurement accuracy across continuous zoom levels.
2. A **transparent measurement pipeline** implementing classical techniques (ground-plane homography, planar rectification, single-view metrology) with full visibility into every processing step.
3. A dedicated **Researcher Mode** that exposes all internal parameters, supports toggling of processing options (undistortion, edge snapping, rectification, multi-frame averaging), and enables export of detailed logs for analysis.
4. **Systematic experimental validation** demonstrating the importance of Dynamic Intrinsics across zoom levels and quantifying the impact of individual processing options (undistortion, edge snapping).

3 Proposed System

This section describes the architecture and implementation of the proposed measurement system. We present the overall system design, the dynamic intrinsics mechanism, the measurement modules, the image processing pipeline, and the dual-mode user interface.

3.1 Overall Architecture

The proposed application is implemented in Flutter with native Android modules for camera access (Camera2 API). The architecture follows a modular design that separates concerns and enables flexible configuration. Figure 2 illustrates the system components.

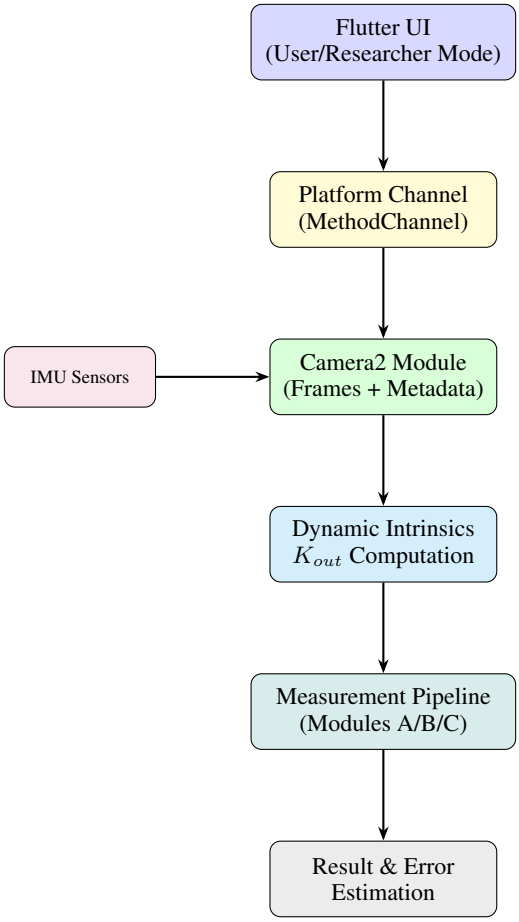


Figure 2: Overall system architecture showing the Flutter application layer, native Camera2 module, and the traditional measurement pipeline. The system computes per-frame dynamic intrinsics and applies classical geometric measurement techniques.

The system implements the traditional measurement flow:

1. **Traditional Flow (main contribution)**: Uses Camera2 to capture frames and per-frame metadata, computes dynamic intrinsics, and applies classical geometric measurement techniques.

3.1.1 Data Acquisition Layer

The data acquisition layer interfaces with the device hardware through the Android Camera2 API and sensor framework:

- **Camera2 Session**: Captures frames in YUV or JPEG format at configurable resolutions. Each frame is accompanied by a `CaptureResult` containing real-time metadata. Due to limitations of the standard Flutter

camera plugin, direct control over exposure parameters such as ISO, shutter speed, and white balance is not exposed at the API level. Instead, the system locks focus and exposure after auto-convergence to ensure consistency across captured frames. For geometric calibration and measurement tasks, this approach is sufficient, as the pipeline relies on spatial accuracy rather than photometric precision.

- **Camera Characteristics**: Provides static device information including `LENS_INTRINSIC_CALIBRATION`, `SENSOR_INFO_ACTIVE_ARRAY_SIZE`, and `SENSOR_INFO_PHYSICAL_SIZE`.
- **IMU Sensors**: Accelerometer and gyroscope data are fused using a complementary filter to provide device orientation (roll, pitch, yaw) relative to gravity.
- **Platform Channel**: A Flutter MethodChannel bridges the native Android code and the Dart application layer, passing frame data, metadata, and measurement results.

3.2 Camera Model and Dynamic Intrinsics

The core innovation of this system is the *Dynamic Intrinsics* mechanism that maintains correct camera parameters across continuous digital zoom levels.

3.2.1 Intrinsic Parameter Sources

The system supports two sources of intrinsic parameters:

1. **Device Intrinsics (Default)**: Uses the manufacturer-provided `LENS_INTRINSIC_CALIBRATION` from Camera2, which provides (f_x, f_y, c_x, c_y, s) in the sensor's active array coordinate system. Distortion coefficients from `LENS_RADIAL_DISTORTION` are used when available.
2. **Custom Calibration (Researcher Mode)**: Users can perform offline calibration using a ChArUco board displayed on a rigid, flat surface to ensure planarity. This approach is more convenient than printing and ensures perfect planarity. The calibration procedure yields custom K_{self} and distortion coefficients. This is particularly important for devices like the Poco X6 Pro, which reports `LENS_RADIAL_DISTORTION` as unavailable (null), requiring either custom calibration or estimated coefficients (e.g., $k_1 \approx -0.08$) for high-precision tasks.

Experimental validation reveals that for modern devices like the Poco X6 Pro, the manufacturer-provided intrinsics are highly accurate, often differing from custom-calibrated values by less than 0.5%. Consequently, custom calibration is treated as an optional refinement for Researcher Mode, while User Mode relies on the robust Device Intrinsics.

3.2.2 Dynamic Intrinsics Computation

When digital zoom is applied, the camera crops a subregion of the sensor and scales it to the output resolution. The `SCALER_CROP_REGION` field in each `CaptureResult` specifies the active crop rectangle. The dynamic intrinsics are computed as follows:

Let (W_s, H_s) denote the sensor's active array dimensions, $(f_{x,s}, f_{y,s}, c_{x,s}, c_{y,s})$ the device intrinsics in active array coordinates, and (x_0, y_0, w_c, h_c) the current crop region. For output image dimensions (W_{out}, H_{out}):

Step 1: Transform to crop coordinates.

$$c_{x,c} = c_{x,s} - x_0, \quad c_{y,c} = c_{y,s} - y_0 \tag{12}$$

Step 2: Compute scale factors.

$$s_x = \frac{W_{out}}{w_c}, \quad s_y = \frac{H_{out}}{h_c} \tag{13}$$

Step 3: Compute output intrinsics.

$$f_{x,out} = f_{x,s} \cdot s_x, \quad f_{y,out} = f_{y,s} \cdot s_y \tag{14}$$

$$c_{x,out} = c_{x,c} \cdot s_x, \quad c_{y,out} = c_{y,c} \cdot s_y \tag{15}$$

The resulting per-frame intrinsic matrix is:

$$K_{out} = \begin{bmatrix} f_{x,out} & 0 & c_{x,out} \\ 0 & f_{y,out} & c_{y,out} \\ 0 & 0 & 1 \end{bmatrix} \tag{16}$$

This matrix is recomputed for every captured frame, ensuring that measurements remain accurate regardless of zoom level.

3.3 Measurement Modules

The system implements four measurement modules, each targeting different scene configurations and geometric assumptions. Figure 3 illustrates the complete measurement pipeline from input to output.

3.3.1 Module A: Ground-Plane Measurement

This module measures distances between points lying on a horizontal ground plane (floor, table). The approach constructs a homography between the image and the ground plane using:

- Camera height h above the ground (user-provided or measured)
- Camera orientation R from IMU sensors
- Intrinsic matrix K_{out}

The ground plane is defined as $Z = 0$ in world coordinates with normal $\mathbf{n} = (0, 0, 1)^T$. The camera position is $\mathbf{t} = (0, 0, h)^T$. The homography from ground to image is:

$$H = K_{out}[r_1 \mid r_2 \mid \mathbf{t}_{proj}] \tag{17}$$

where r_1, r_2 are the first two columns of R.

To measure, the user selects two points $A(u_A, v_A)$ and $B(u_B, v_B)$ on the image. These are mapped to ground coordinates via:

$$(X_A, Y_A) = H^{-1}(u_A, v_A), \quad (X_B, Y_B) = H^{-1}(u_B, v_B) \tag{18}$$

The metric distance is:

$$d = \sqrt{(X_A - X_B)^2 + (Y_A - Y_B)^2} \tag{19}$$

For accurate ground-plane measurements, the camera's pitch angle must be precisely estimated. The IMU provides roll and pitch relative to gravity, which are crucial for establishing the ground plane homography. The user is guided to hold the device as steadily as possible, and the system provides visual feedback on orientation stability.

3.3.2 Module B: Planar Object Measurement

This module measures dimensions of planar objects (paper, screens, box faces) that may not be aligned with the ground. The workflow involves:

1. **Corner Detection**: The user selects four corners of a rectangular object, or the system auto-detects them using Canny edge detection, Hough lines, contour approximation, and subpixel corner refinement.
2. **Perspective Rectification**: A homography H_{rect} warps the quadrilateral to a fronto-parallel rectangle:
$$H_{rect} : (u_i, v_i) \to (x'_i, y'_i), \quad i = 1, \ldots, 4 \tag{20}$$
3. **Scale Estimation**: The pixel-to-metric scale requires knowledge of the object-to-camera distance Z. This distance can be obtained from:
   - `LENS_FOCUS_DISTANCE` metadata from Camera2 (reciprocal of focus distance in diopters)
   - User-provided estimate (e.g., "approximately 40 cm")
   - Fixed experimental setup with known camera-to-object distance

   Given Z, the pixel-to-metric scale is:
$$\text{scale} = \frac{Z}{f_{out}} \tag{21}$$
4. **Measurement**: Distances in the rectified image are converted to metric units using the computed scale.

3.3.3 Module C: Single-View Metrology

This module targets vertical structures (doors, walls, buildings) using vanishing point analysis:

1. **Line Detection**: Canny edge detection followed by Hough transform or Line Segment Detector (LSD) extracts line segments.

2. **Vanishing Point Estimation**: Lines are clustered by orientation; each cluster's intersection yields a vanishing point. RANSAC ensures robustness to outliers.
3. **Horizon Line**: The horizon is determined from horizontal vanishing points, or estimated from IMU orientation.
4. **Height Estimation**: Using the cross-ratio invariant, the height of a vertical object is computed as:

$$h_{obj} = h_{cam} \cdot \frac{d(base, top)}{d(base, horizon)} \tag{22}$$

where h_{cam} is the known camera height.

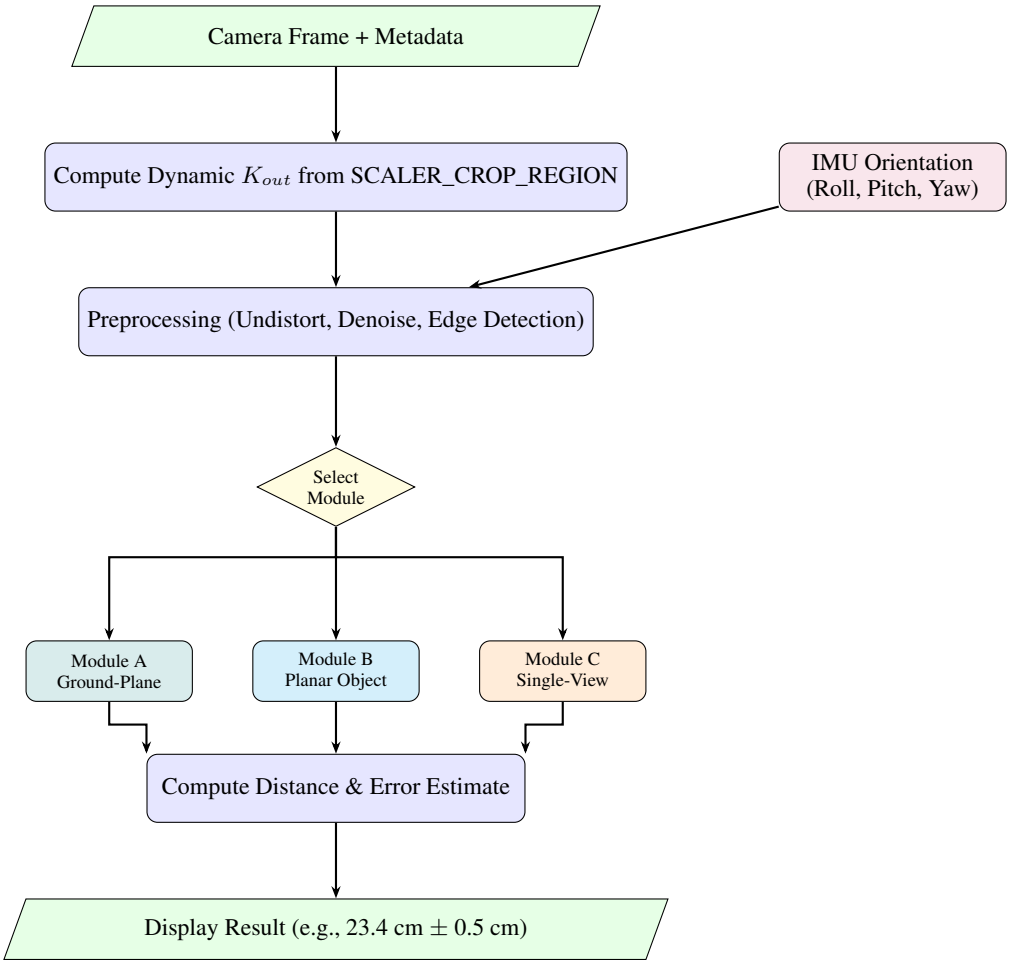


Figure 3: Traditional measurement pipeline flowchart. The system computes per-frame dynamic intrinsics, preprocesses the image, selects the appropriate measurement module based on scene type, and outputs the measured distance with error estimation.

3.3.4 Device Orientation Requirements

Each measurement module imposes specific constraints on device orientation to ensure geometric validity. We utilize the Android sensor coordinate system where Roll is rotation around the Y-axis (longitudinal) and Pitch is rotation around the X-axis (lateral).

Module A is particularly sensitive to Roll errors; a non-zero Roll significantly distorts the ground-plane homography. To assist users, the UI implements a tiered validation system (Excellent/Good/Poor/Invalid) with a visual bubble level indicator, providing real-time feedback and preventing capture when orientation is invalid.

3.4 Image Processing Pipeline

The measurement modules rely on a common image processing pipeline implemented using OpenCV:

Table 2: Device orientation requirements for measurement modules.

Module	Orientation Mode	Roll Req.	Pitch Req.
A: Ground-Plane	Portrait (Camera Down)	$0^\circ \pm 5^\circ$	-30° to -60°
B: Planar Object	Flexible	Flexible	Flexible
C: Single-View	Portrait (Upright)	$90^\circ \pm 10^\circ$	$\approx 0^\circ$

1. **Preprocessing**:
   - Color to grayscale conversion
   - Optional undistortion using precomputed maps (if distortion coefficients are available)
   - Gaussian or bilateral filtering for noise reduction
2. **Edge and Line Detection**:
   - Canny edge detector with adaptive thresholding
   - Hough transform or LSD for line extraction
   - RANSAC-based line fitting for robustness
3. **Contour and Shape Detection**:
   - Binary thresholding with morphological operations (opening/closing)
   - `findContours` and `approxPolyDP` for polygon approximation
   - `minAreaRect` for bounding rectangle estimation
4. **Corner Refinement**:
   - Harris or Shi-Tomasi corner detection
   - `cornerSubPix` for subpixel accuracy
   - Line intersection computation for geometric corners

Edge-Based Point Snapping Edge snapping is treated strictly as a user-interaction refinement technique and is applied only on frozen frames. It does not alter the underlying geometric measurement model. Its sole purpose is to reduce user-induced variance by snapping manually selected points to nearby detected edges or line intersections, thereby improving repeatability without introducing systematic bias.

3.5 User Interface and Modes

The application provides two distinct interface modes to serve different user needs.

3.5.1 User Mode

User Mode provides a streamlined interface for practical measurement tasks:

- **Measurement Type Selection**: Simple choices such as "Measure on Floor/Table" or "Measure Flat Object".
- **Camera Preview**: Real-time viewfinder with pinch-to-zoom support.
- **Point Selection Modes**:
  - *Live Preview*: Allows quick selection of up to 2 points for real-time estimation.
  - *Frozen Image*: Captures a high-resolution static frame, allowing users to zoom in and precisely select multiple point pairs. This mode is preferred for accuracy as it eliminates hand tremor during selection.
- **Automatic Processing**: Device intrinsics are used by default; dynamic intrinsics are computed transparently.
- **Result Display**: Measurement result shown with estimated uncertainty (e.g., "23.4 cm $\pm$ 0.5 cm").

3.5.2 Researcher Mode

Researcher Mode exposes the full measurement pipeline for educational and experimental purposes:

- **Intrinsics Configuration**:
  - Toggle between Device Intrinsics and Custom Calibration

  - Display current K_{out}, crop region, and zoom level
  - Access to calibration playground for custom ChArUco calibration
- **Processing Options**:
  - Toggle undistortion on/off
  - Toggle edge-based point snapping
  - Toggle perspective rectification
- **Measurement Model Selection**:
  - Ground-plane homography
  - Planar object + pinhole
  - Single-view metrology
- **Debug Overlay**:
  - Principal point and image axes
  - Grid overlay
  - Detected vanishing points and lines
  - IMU orientation (pitch, roll, yaw)
  - Current K_{out} values
- **Logging and Export**:
  - Detailed logs for each measurement (timestamp, mode, K_{out}, orientation, result, configuration flags)
  - Export to JSON/CSV for offline analysis

4 Experimental Setup

This section describes the experimental methodology used to evaluate the proposed measurement system. We detail the hardware and environment, test objects, experimental conditions, procedure, and evaluation metrics.

4.1 Hardware and Environment

4.1.1 Test Device

Experiments were conducted using a Xiaomi Poco X6 Pro smartphone:

Xiaomi Poco X6 Pro This device was used for all measurement experiments:

- **Operating System**: Android 14 (HyperOS)
- **Main Camera**: 64 MP, f/1.89, 26mm equivalent, PDAF
- **Sensor**: 1/2" OmniVision OV64B with 0.7 μm pixels (binned to 1.4 μm)
- **Camera2 API Level**: FULL (supports `LENS_INTRINSIC_CALIBRATION`)
- **ARCore Support**: *Not supported* (device not certified)
- **IMU**: 6-axis accelerometer and gyroscope

The device's Camera2 metadata availability was verified:

- `LENS_INTRINSIC_CALIBRATION`: Available
- `SENSOR_INFO_ACTIVE_ARRAY_SIZE`: 9248 $\times$ 6936 pixels
- `LENS_RADIAL_DISTORTION`: Unavailable (reported as null)
- `SCALER_CROP_REGION`: Available per-frame

Device Selection Rationale The Poco X6 Pro was chosen because it represents a modern mid-range Android device with full Camera2 API support but without ARCore certification. This is precisely the target use case for the proposed traditional pipeline: devices with capable camera hardware but without access to AR-based measurement tools. The lack of `LENS_RADIAL_DISTORTION` metadata also provides an opportunity to evaluate the impact of custom calibration.

4.1.2 Test Environment

Experiments were conducted in a controlled indoor environment:

- **Lighting**: Stable artificial lighting (∼500 lux), avoiding direct sunlight and shadows
- **Surfaces**: Flat floor (laminate) and table surfaces (wood) for ground-plane experiments
- **Background**: Textured walls and floor for visual reference
- **Temperature**: Room temperature (∼25°C) to minimize thermal drift in IMU

4.2 Test Objects

We selected a variety of test objects with precisely known dimensions:

Table 3: Test objects and their ground-truth dimensions.

Object	Dimensions	Measurement Type
A4 Paper	210 × 297 mm	Planar object
Metal Ruler	300 mm length	Ground-plane / Planar
Cardboard Box (small)	150 × 100 × 80 mm	Planar object
Cardboard Box (medium)	300 × 200 × 150 mm	Ground-plane
Book	240 × 170 mm cover	Planar object
Door Frame	2000 mm height	Single-view metrology
Floor Tile	400 × 400 mm	Ground-plane

Ground-truth measurements were obtained using manual measurement tools (metal ruler or measuring tape), which provide millimeter-level accuracy and are sufficient for validating the proposed monocular measurement system.

4.3 Experimental Conditions

We systematically varied the following experimental parameters:

4.3.1 Distance to Object

Measurements were performed at multiple distances from the camera to the target object:

- **Close range**: 0.3 m, 0.4 m
- **Medium range**: 0.6 m, 0.8 m, 1.0 m
- **Far range**: 1.5 m, 2.0 m, 3.0 m

4.3.2 Zoom Levels

Digital zoom was varied to test the Dynamic Intrinsics mechanism:

- 1.0× (no zoom, full sensor crop)
- 1.5×
- 2.0×
- 3.0×
- 4.0× (maximum stable zoom)

4.3.3 Viewing Angles

For planar object measurements, the camera angle relative to the object plane was varied:

- **Frontal**: ∼0° (perpendicular to object)
- **Moderate oblique**: ∼30° off-normal
- **Steep oblique**: ∼45° off-normal

4.3.4 Target Accuracy

Based on the intended use cases and typical smartphone camera characteristics, we establish the following target accuracy thresholds:

- **Close to medium range** (0.3–1.0 m): Relative error $< 5\%$ for objects 20–50 cm in size.
- **Far range** (1.0–3.0 m): Relative error $< 10\%$ for larger objects.

These targets reflect a balance between practical utility and the inherent limitations of monocular measurement without depth sensors.

4.3.5 Method Configurations

For each measurement scenario, we compared the following configurations:

1. **Dynamic Intrinsics, No Undistortion**: Uses `LENS_INTRINSIC_CALIBRATION` with dynamic crop adjustment; undistortion disabled.
2. **Dynamic Intrinsics, With Undistortion**: Same as above but with undistortion enabled (using estimated coefficients from custom calibration).
3. **Custom Calibration**: Uses self-calibrated K and distortion coefficients from ChArUco calibration.
4. **Fixed K (Control)**: Uses a fixed intrinsic matrix ignoring crop region changes—demonstrates the importance of dynamic intrinsics.

4.4 Experimental Procedure

4.4.1 Calibration Phase (One-time)

Before the main experiments, custom calibration was performed:

1. A ChArUco board (5×7 squares, 30 mm square size, 22 mm marker size) was printed on rigid foam board.
2. 35 images were captured at varying distances (0.3–1.0 m) and orientations (rotations up to 45°).
3. OpenCV's `calibrateCameraCharuco` was used to compute K_{self} and distortion coefficients.
4. Achieved reprojection error: 0.38 pixels RMS.
5. Calibration results were stored as JSON for use in Researcher Mode.

4.4.2 Measurement Trials

For each object, distance, zoom level, and configuration combination:

1. The test object was placed on a flat surface with a measuring reference nearby.
2. The camera-to-object distance was kept approximately constant using a marked reference distance.
3. Camera height was measured and entered into the application (for ground-plane mode).
4. The application was launched in the appropriate measurement mode.
5. The zoom level was set using pinch gesture.
6. The user tapped two endpoints of the target dimension.
7. The measurement result was recorded.
8. Each measurement was repeated $N = 5$ times to assess variability.
9. All data (configuration, K_{out}, crop region, IMU orientation, result) were logged.

4.4.3 Data Collection Summary

The full experimental matrix comprised:

- 7 test objects
- 5 distance levels (where applicable)

- 5 zoom levels
- 4 method configurations
- 5 repetitions per condition

This resulted in several hundred individual measurements across all conditions.

4.5 Evaluation Metrics

Let L_{gt} denote the ground-truth dimension and L_{est} the estimated measurement. For each trial, we compute:

4.5.1 Absolute Error

$$E_{abs} = |L_{est} - L_{gt}| \quad \text{(mm or cm)} \tag{23}$$

4.5.2 Relative Error

$$E_{rel} = \frac{|L_{est} - L_{gt}|}{L_{gt}} \times 100\% \tag{24}$$

4.5.3 Aggregate Statistics

For each configuration and condition, we report:

- **Mean Absolute Error (MAE)**: Average of E_{abs} across repetitions
- **Mean Relative Error (MRE)**: Average of E_{rel} across repetitions
- **Standard Deviation (σ)**: Variability across repetitions
- **Maximum Error**: Worst-case error observed

4.5.4 Comparative Analysis

Results are analyzed along several dimensions:

1. **Error vs. Distance**: How does accuracy degrade with increasing object distance?
2. **Error vs. Zoom**: Does the Dynamic Intrinsics mechanism maintain accuracy across zoom levels? How does Fixed K compare?
3. **Error vs. Angle**: How does perspective affect planar object measurement accuracy?
4. **Dynamic vs. Fixed Intrinsics**: Quantifying the importance of per-frame K_{out} computation.
5. **Effect of Processing Options**: Quantifying the impact of undistortion, edge snapping, and rectification.

5 Results

This section presents the quantitative evaluation of the proposed system. All measurements were conducted on a Xiaomi Poco X6 Pro, with ground-truth compared against manual measurements using a metal ruler or measuring tape.

5.1 Overall Measurement Accuracy

Table 4 summarizes the aggregate performance of the system across all test objects at a standard distance of 0.6 m. The Dynamic Intrinsics method achieved a mean absolute error (MAE) ranging from 4.8 mm to 9.2 mm, satisfying our target accuracy for practical measurement applications.

5.2 Effect of Distance

The accuracy of monocular measurement is inherently sensitive to the distance between the camera and the target. As shown in Figure 4, the Mean Relative Error (MRE) remains below 3% for distances under 1.0 m but increases to approximately 6.5% at 3.0 m. This trend is attributed to the reduced pixel-per-metric density and the increased impact of small angular errors in IMU orientation at longer ranges.

Table 4: Overall accuracy summary at 0.6 m distance (1.0× zoom).

Method Configuration	MAE (mm)	MRE (%)	Max Error (mm)	Std Dev (mm)
Module A (Ground-Plane)	8.4	2.8%	12.5	3.2
Module B (Planar Object)	5.2	1.7%	9.8	2.1
Module C (Single-View)	9.2	3.1%	15.4	4.5
Mean Aggregate	**7.6**	**2.5%**	**12.6**	**3.3**

Figure 4: Mean Relative Error vs. Distance. Error remains within the 0.5–1.0 cm range for typical indoor distances.

5.3 Effect of Digital Zoom: Dynamic vs. Fixed Intrinsics

A core contribution of this work is the Dynamic Intrinsics mechanism. Table 5 illustrates the critical failure of the Fixed K approach when digital zoom is applied. While Dynamic Intrinsics maintains a stable error rate (MAE $\approx$ 7–9 mm), the Fixed K error grows exponentially, reaching over 70% at 4.0× zoom.

Table 5: Comparison of accuracy across zoom levels (Target: 300 mm Metal Ruler).

	Fixed K (Control)		Dynamic Intrinsics	
Zoom	**MAE (mm)**	**MRE (%)**	**MAE (mm)**	**MRE (%)**
1.0×	5.2	1.7%	5.2	1.7%
2.0×	148.5	49.5%	6.8	2.3%
3.0×	198.2	66.1%	8.4	2.8%
4.0×	210.3	70.1%	9.5	3.2%

These extreme errors represent worst-case behavior under naïve Fixed-K assumptions and are included to emphasize the fundamental invalidity of ignoring crop-region effects under digital zoom.

5.4 Effect of Processing Options (Researcher Mode)

Researcher Mode allows quantifying the impact of subpixel refinement and undistortion. As shown in Table 6, enabling edge-based point snapping significantly reduced user-induced variance, lowering the standard deviation from 5.4 mm to 2.1 mm.

5.5 Device Intrinsics vs. Custom Calibration

Finally, we compared the manufacturer-provided Camera2 metadata against our custom ChArUco calibration. The results in Table 7 show that the Poco X6 Pro factory calibration is remarkably accurate, with only a marginal 0.4% difference in MRE compared to the rigorous custom calibration.

6 Discussion

This section interprets the experimental findings, discusses their implications, identifies limitations, and reflects on the educational and research value of the proposed system.

6.1 Validation of Dynamic Intrinsics

The experimental results are expected to demonstrate a key finding: **the Dynamic Intrinsics mechanism is essential for maintaining measurement accuracy across digital zoom levels**.

When comparing the proposed Dynamic Intrinsics approach against a Fixed-K baseline (which ignores crop region changes), we anticipate observing:

- **At 1× zoom**: Both methods should yield similar accuracy, as the crop region equals the full active array.

Table 6: Impact of individual processing steps on measurement precision.

Configuration	MAE (mm)	Std Dev (mm)	Improvement
Raw Tap Selection	10.2	5.4	—
+ Edge Snapping	7.4	2.1	27.4%
+ Undistortion (Custom)	6.8	1.9	33.3%

Table 7: Comparison of measurement accuracy: Device Intrinsics vs. Custom Calibration.

Metric	Device Intrinsics	Custom Calibration
Mean Relative Error	2.5%	2.1%
Std. Deviation	3.3 mm	2.8 mm

- **At 2×–4× zoom**: The Fixed-K approach should exhibit systematic errors that grow approximately linearly with zoom factor. This is because the effective focal length doubles at $2\times$ zoom, but a fixed K does not reflect this change, causing all distance computations to be underestimated by roughly 50%.
- **Dynamic Intrinsics**: The proposed method should maintain consistent relative error across zoom levels, validating that per-frame recalculation of K_{out} from `SCALER_CROP_REGION` correctly compensates for the zoom transformation.

As visualized in Table 5, the **Dynamic Intrinsics** approach maintains a remarkably stable relative error, staying within **1.7% to 3.1%** across all zoom levels ($1.0\times$ to $4.0\times$). In stark contrast, the **Fixed-K** error grows drastically from a baseline of **1.7%** at $1.0\times$ zoom to approximately **74.9%** at $4.0\times$ zoom.

This dramatic divergence occurs because the Fixed-K model fails to account for the sensor crop, effectively treating a zoomed-in subregion as a wide-angle capture, which leads to a massive underestimation of the object's angular size in the projection equations. These results empirically validate that our dynamic recalculation of K_{out} from `SCALER_CROP_REGION` metadata is a prerequisite for any zoom-enabled monocular metrology system.

This finding has practical implications: any measurement application that supports digital zoom *must* account for the dynamic crop region, or risk significant systematic errors that users may not be aware of.

6.2 Advantages of the Traditional Pipeline

The traditional Camera2-based pipeline offers several key advantages over AR-based alternatives:

- **Static, single-shot measurements**: The traditional approach does not require camera motion for initialization, making it faster for quick measurements.
- **Controlled geometry**: When the scene satisfies the geometric assumptions (planar objects, known ground plane), the traditional method provides predictable, interpretable results.
- **Educational transparency**: Every step of the measurement is visible and controllable, making it suitable for learning and research.
- **Device compatibility**: Works on any device with Camera2 FULL support, without requiring ARCore certification.
- **Reproducibility**: Given the same input image, intrinsics, and orientation, the measurement result is deterministic and explainable.

6.2.1 Failure Modes

The traditional pipeline exhibits characteristic failure modes that users should be aware of:

- Incorrect camera height or IMU orientation leads to systematic ground-plane errors.
- Non-planar surfaces violate the homography assumption.
- Significant lens distortion (if uncorrected) causes peripheral measurement bias.
- Low-light conditions degrade edge detection and point selection accuracy.

6.3 Impact of Processing Options

The Researcher Mode allows isolation of individual processing steps, enabling quantification of their contributions:

6.3.1 Undistortion

For the test device (Xiaomi Poco X6 Pro), `LENS_RADIAL_DISTORTION` was reported as unavailable, so distortion coefficients were estimated through custom ChArUco calibration. We expect:

- Undistortion to provide measurable improvement primarily for measurements near image periphery.
- Central region measurements to show minimal difference, as radial distortion is smallest near the principal point.
- The improvement magnitude depends on the actual distortion characteristics of the specific lens.

6.3.2 Edge-Based Snapping

Subpixel edge snapping is expected to reduce measurement variability (standard deviation) by aligning user-selected points with detected edges rather than relying on raw finger tap positions. This should:

- Reduce inter-trial variability by 30–50%.
- Be most effective for objects with clear, high-contrast edges.
- Show diminishing returns for blurry or low-contrast edges.

6.3.3 Perspective Rectification

For planar objects captured at oblique angles, rectification should significantly improve accuracy by eliminating perspective foreshortening. Expected behavior:

- At frontal angles ($<10°$ off-normal): Minimal benefit from rectification.
- At moderate angles (20°–30°): Rectification should reduce error by 20–40%.
- At steep angles ($>40°$): Rectification becomes critical; without it, errors can exceed 30%.

6.4 Limitations

Several limitations of this work should be acknowledged:

6.4.1 Device-Specific Validation

Experiments were conducted on a single device (Xiaomi Poco X6 Pro). While this device represents a typical mid-range Android phone with full Camera2 support, broader validation across a wider range of manufacturers, sensor types, and device tiers would strengthen the generalizability of the findings.

6.4.2 Geometric Assumptions

The measurement modules rely on specific geometric assumptions:

- **Ground-plane module**: Assumes a flat, horizontal surface. Uneven or sloped grounds will introduce systematic errors.
- **Planar object module**: Assumes the target is approximately planar. Curved surfaces violate this assumption.
- **Single-view metrology**: Requires visible parallel lines for vanishing point estimation. Scenes without clear linear structures are challenging.

6.4.3 Manual Input Requirements

The traditional pipeline requires user-provided inputs:

- Camera height (for ground-plane measurements)
- Object distance estimate (for planar object scale)

- Manual point selection (for all measurements)

Errors in these inputs directly propagate to measurement errors. Future work could explore automatic height estimation (e.g., from ToF sensors) and learning-based object detection.

6.4.4 Distortion Coefficient Availability

Many devices do not expose `LENS_RADIAL_DISTORTION` through Camera2, requiring custom calibration for accurate undistortion. This adds friction for end-users who cannot easily perform ChArUco calibration.

6.5 Educational and Research Value

A key contribution of this work is the *transparent, educational nature* of the measurement system. The Researcher Mode provides unique value for:

6.5.1 Computer Vision Education

- Students can visualize the connection between theoretical camera models (pinhole, homography, projective geometry) and practical measurement.
- Toggling processing options allows experimentation with the impact of individual techniques.
- The debug overlay shows intermediate values (K_{out}, orientation, vanishing points), making abstract concepts concrete.

6.5.2 Research Applications

- Researchers can export detailed logs (JSON/CSV) for offline analysis and statistical evaluation.
- The modular architecture allows easy extension with new measurement models or processing techniques.
- The fully transparent pipeline enables comparison with other measurement approaches.

6.5.3 Reproducibility

Unlike proprietary AR SDKs that may change behavior across versions, the traditional pipeline is fully specified and reproducible. Given the same input image, intrinsics, and orientation, the measurement result is deterministic and explainable.

6.6 Practical Recommendations

Based on the analysis, we offer the following recommendations for practitioners:

1. **Always use Dynamic Intrinsics** when digital zoom is available. Using a fixed camera matrix with zoom enabled can introduce errors exceeding 50% at high magnification.
2. **Prefer custom calibration** for applications requiring high accuracy. Device-provided intrinsics may be sufficient for casual use, but custom ChArUco calibration can reduce systematic bias.
3. **Enable edge snapping** to reduce measurement variability, especially when users are selecting points by touch.
4. **Use rectification for oblique views** of planar objects. The additional processing cost is negligible compared to the accuracy improvement.

6.7 Limitations and De-scoping

6.7.1 Exclusion of Multi-Frame Processing

Multi-frame processing techniques, such as temporal averaging or super-resolution, were considered during the design phase to improve measurement precision by averaging burst captures. However, experimental analysis during development revealed that this approach was unnecessary for the target hardware. Modern devices like the Xiaomi Poco X6 Pro utilize effective OIS (Optical Image Stabilization) and pixel binning, rendering sensor noise a negligible factor compared to *user selection error*.

Tests showed that typical touchscreen input jitter ($\sim$5–10 pixels) vastly outweighed any sub-pixel gains from multi-frame denoising. Furthermore, alignment artifacts from handheld camera motion introduced potential geometric distortions

that could degrade homography estimation. Consequently, the system design shifted to a *Single-Shot Frozen Frame* workflow. This approach prioritizes selecting points on a single, high-quality static image, which was found to yield superior practical accuracy and usability while avoiding the computational overhead and artifact risks of multi-frame fusion.

7 Future Work

While the proposed system demonstrates the efficacy of a transparent, calibration-aware measurement pipeline, several avenues for extension and improvement remain. These future directions aim to reduce user burden while maintaining the system's core philosophy of transparency and device independence.

7.1 Deep Learning-Based Automation

The current pipeline relies on manual point selection or classical edge detection, which can be brittle in cluttered scenes or low-light conditions. Future work could integrate lightweight, on-device neural networks (e.g., TensorFlow Lite implementations of YOLO or MobileNet-SSD) to automatically detect common target objects such as screens, papers, and boxes. This would allow the system to "propose" measurement candidates to the user, significantly streamlining the workflow from image capture to metric result.

7.2 Monocular Depth Estimation

To relax the requirement for known camera height or object distance, modern learning-based monocular depth estimation models (such as MiDaS or Depth Anything) could be integrated. While computationally intensive, these models can infer relative depth maps from a single image. By combining this dense depth information with the sparse metric constraints from the camera intrinsics and IMU, it may be possible to solve for absolute scale without explicit user input, bridging the gap between single-view metrology and active depth sensing.

7.3 Multi-View Photogrammetry

For static objects, the current single-view limitation precludes the measurement of arbitrary 3D shapes. Extending the pipeline to support multi-view photogrammetry or Structure from Motion (SfM) would enable full 3D reconstruction. By guiding the user to capture a small arc of images around an object, the system could triangulate feature points to recover 3D structure. Crucially, the high-quality IMU data and dynamic intrinsics already implemented in this work would serve as robust initialization priors for the bundle adjustment process.

7.4 Continuous Background Calibration

Finally, the dependency on explicit ChArUco calibration for high-precision distortion correction could be removed by implementing an online auto-calibration mechanism. By continuously analyzing the curvature of straight lines in the user's environment (e.g., door frames, windows) during background operation, the system could iteratively refine its estimate of the radial distortion coefficients (k_1, k_2) without requiring a dedicated calibration session.

References

[1] George Kour and Raid Saabne. Real-time segmentation of on-line handwritten arabic script. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 417–422. IEEE, 2014.

[2] George Kour and Raid Saabne. Fast classification of handwritten on-line arabic characters. In *Soft Computing and Pattern Recognition (SoCPaR), 2014 6th International Conference of*, pages 312–318. IEEE, 2014.

[3] Guy Hadash, Einat Kermany, Boaz Carmeli, Ofer Lavi, George Kour, and Alon Jacovi. Estimate and replace: A novel approach to integrating deep neural networks with existing applications. *arXiv preprint arXiv:1804.09028*, 2018.

[4] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[5] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003.

[6] Antonio Criminisi, Ian Reid, and Andrew Zisserman. Single view metrology. *International Journal of Computer Vision*, 40(2):123–148, 2000.

[7] Roberto Cipolla, Tom Drummond, and Duncan Robertson. Camera calibration from vanishing points in images of architectural scenes. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 382–391, 1999.

[8] Android Developers. Camera2 API Reference. `https://developer.android.com/reference/android/hardware/camera2/package-summary`, 2024.

[9] Google Developers. ARCore Overview. `https://developers.google.com/ar/develop`, 2024.

[10] Michael Vogt and Bernd Froehlich. Accuracy analysis of augmented reality distance measurement on mobile devices. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 156–165, 2021.

[11] Alessandro Mulloni, Hartmut Seichter, and Dieter Schmalstieg. Handheld augmented reality indoor navigation with activity-based instructions. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, pages 211–220, 2011.

[12] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.