RUNE-Tag: a High Accuracy Fiducial Marker with Strong Occlusion Resilience

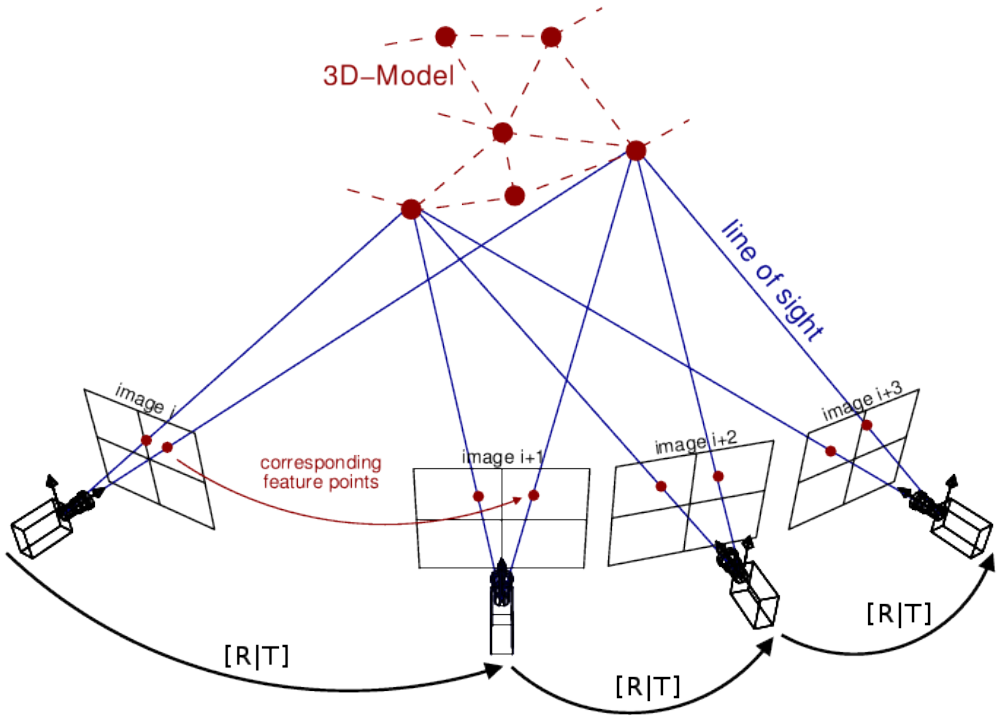
Filippo Bergamasco, Andrea Albarelli, Emanuele Rodolá and Andrea Torsello

Dipartimento di Scienze Ambientali, Informatica e Statistica
Università Ca'Foscari di Venezia

May 17, 2012

Introduction

Pose estimation



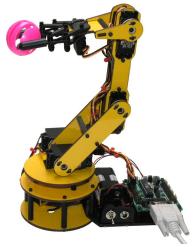
Find the transformation needed to map an object model from its inherent coordinate system into agreement with the sensory data (cameras, in our case) [4]

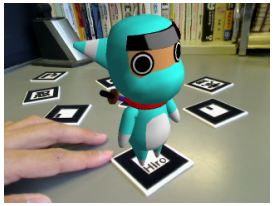
Introduction

Pose estimation - Applications

Is a crucial task for many computer vision task:

- 3D scene reconstruction.
- Object grasping, manipulation and recognition
- Augmented reality
- Photogrammetry
- etc.

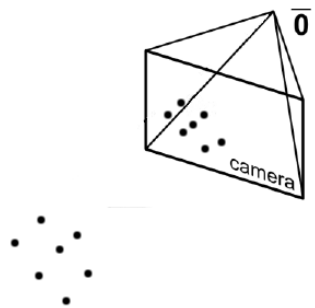




Introduction

2D-3D pose estimation problem

We are interested to find the rigid motion that best fits 2d object data (as seen by cameras) to the 3D model of that data.
The camera is calibrated (known focal length and principal point).



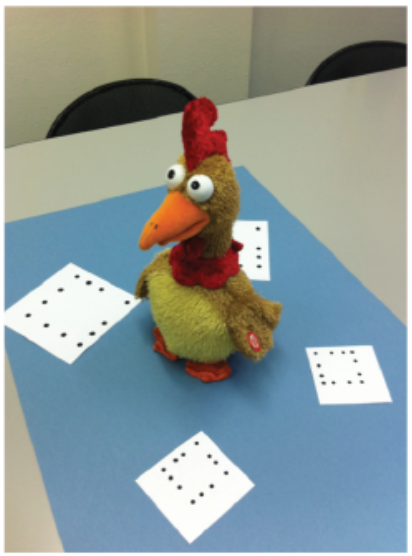
Many approaches has been proposed in literature. Many of them based on a non-linear optimization process that attempts to minimize the re-projection error of 3D model with respect to the data.

Introduction

2D-3D pose estimation

How to find these correspondences?

From interest points
found in image itself



From artificial features
placed into the scene
(fiducial markers)

Introduction

Fiducial Markers

- Fiducial markers are widely adopted tools to add reliable model-based features into an otherwise general scene.
- Are the weapon of choice when high accuracy or real-time pose estimation must be performed

Many of them proposed in literature [3, 6, 5, 2]:

Introduction

Our Goals

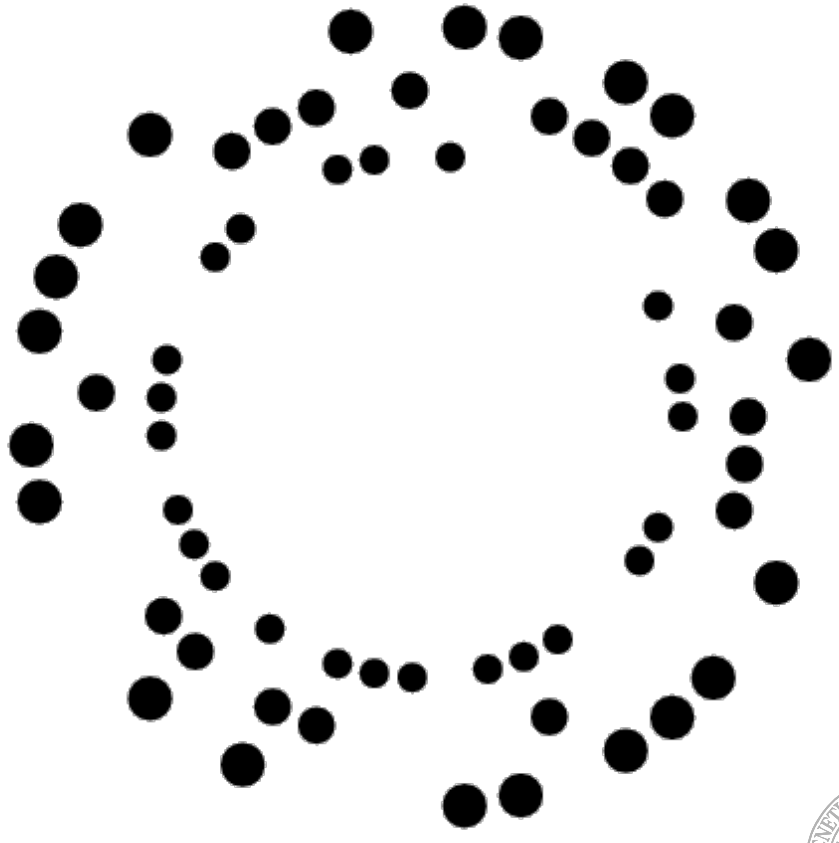
We designed a novel class of fiducial marker design with the following goals in mind:

- Very accurate pose estimation (good coarse registration for 3d scanning)
- Take advantage of the same basic features for both detection and recognition (no image rectification)
- Allow a large number of different tags
- Robust to severe occlusions
- Leave some free payload space for additional user content

RUNE-Tag Design

Multiple levels

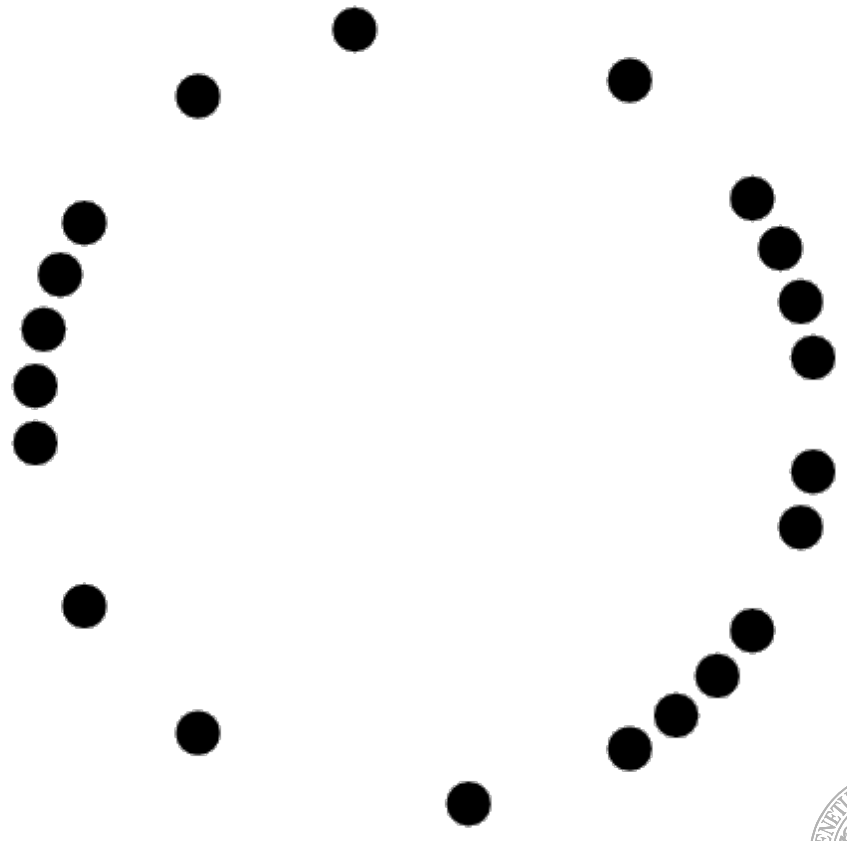
- Set of n concentric rings called **levels**
- Each ring divided in m evenly distributed **sectors**
- Each **slot** (level,sector) can have a dot
- Dot size depends by its level



RUNE-Tag Design

Single level

- Set of *n* concentric rings called **levels**
- Each ring divided in *m* evenly distributed **sectors**
- Each **slot** (level,sector) can have a dot
- Dot size depends by its level



RUNE-Tag Detection

Some general observations

The whole idea behind tag detection is that the ellipse class is invariant with respect to projective transformations.

- Each dot is an ellipse and will remain an ellipse from any point of view
- For the whole tag the same idea holds

However...

- It is still difficult to cluster all ellipses belonging to each tag
- A simple RANSAC scheme will be unfeasible

RUNE-Tag Detection

Why an extensive search is unfeasible?

We require at least 5 points to define an ellipse with no ambiguity.

Naive approach:

1. For each group of 5 dots:
2. Fit an ellipse on all the centres
3. See how many other dots falls near this ellipse and mark them
4. Check if we have found a valid tag

In practice, the problem cannot be solved this way:

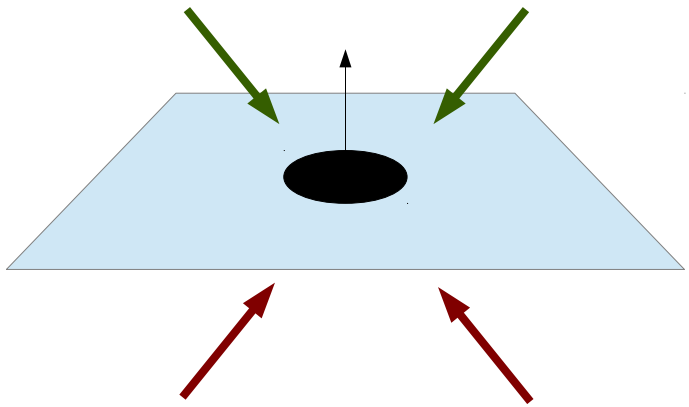
- With 100 ellipses detected, more than $70 * 10^6$ groups should be tested.

RUNE-Tag Detection

Key observation

The projective transformation that affects each dot in a ring is the same of the one that transforms the ring itself.

- By observing an ellipse found in a scene, if we assume that belongs to our model, we can found the rotation(s) that could have generated that transformation.

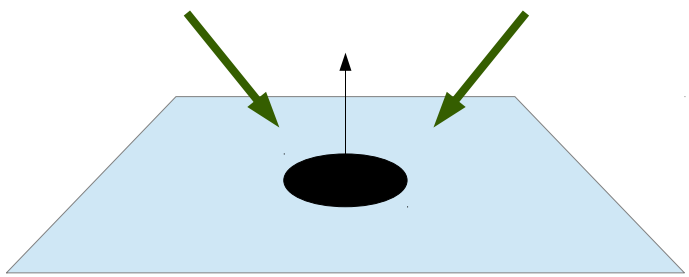


RUNE-Tag Detection

Key observation

The projective transformation that affects each dot in a ring is the same of the one that transforms the ring itself.

- By observing an ellipse found in a scene, if we assume that belongs to our model, we can found the rotation(s) that could have generated that transformation.



Two of them can be discarded because incompatible with plane normal. (We cannot see the back of the tag)

RUNE-Tag Detection

Step 1

Ellipse detection:

1. Adaptive threshold
2. Find contours
3. Ellipse fitting
4. Filtering

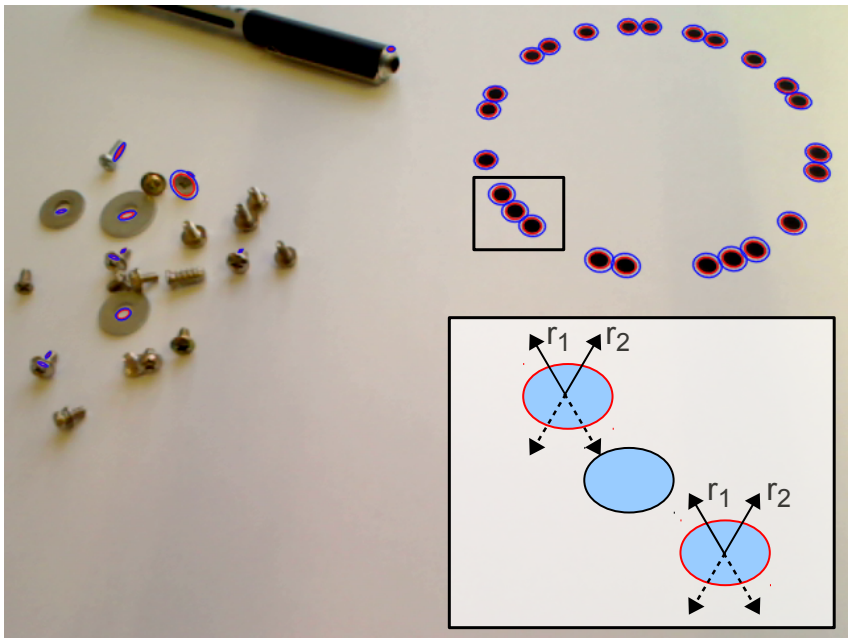
Optional:

- Filtered ellipses refined as proposed in [7]. (Slow, but lead to better precision)

RUNE-Tag Detection

Step 2

Following [1], all pairs of feasible view directions are computed for each ellipse detected in the scene

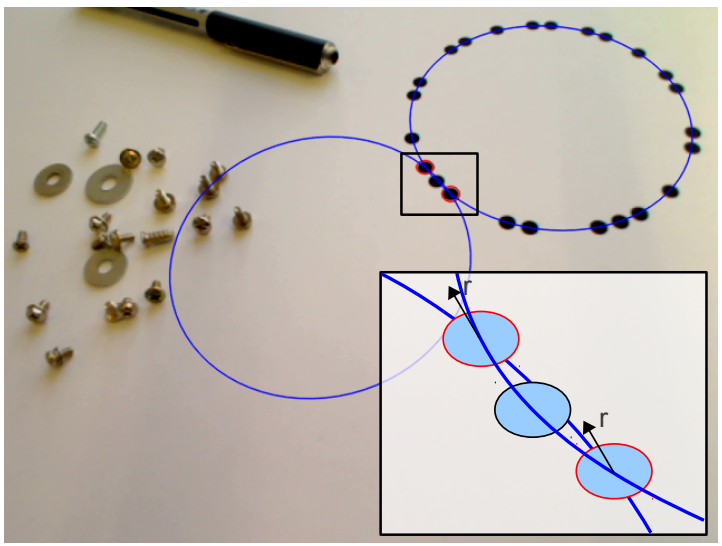


More details on that later...

RUNE-Tag Detection

Step 3

For each **pair** of ellipses, we are now able to estimate the two possible rings in which they perhaps belong:

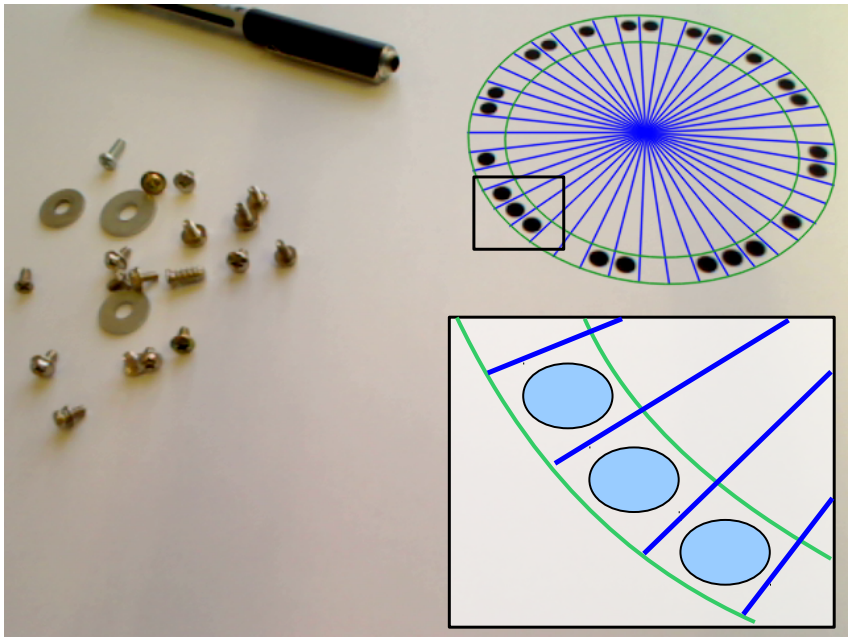


- We chose the "best" pair of rotations and average it.
- Because a known proportion of radii between the ring and all dots composing the ring, we can estimate the two ring circles
- Ring circles can be transformed back to get two tag guesses

RUNE-Tag Detection

Step 4

The guessed tags are divided in slots and a **code** is generated based on the presence (or absence) of dots inside each of those



The recognition step will then determine if the tag is valid and identify each dot of the tag for pose estimation.

(More on that later)

RUNE-Tag Detection

Multi-level tags

Multi-level tags are detected essentially in the same way, with some additional concerns:

- All pairs are filtered with respect to radii difference (we don't want to process pairs of ellipses belonging to different levels).
- We can't know at which level the two ellipses used to fit the initial ring belong. All possibilities have to be considered and checked in the recognition step.

RUNE-Tag Detection

Ellipse rotation estimation

If a circle is projected on to the image plane with perspective projection, it shows an ellipse in general case.

- From this single conic is not possible to recover the full camera pose, but we can estimate the rotation(s) around the optical center that transforms that ellipse in a circle

An ellipse in the image plane can be described by the following equation:

$$Ax_e^2 + 2Bx_e y_e + Cy_e^2 + 2Dx_e + 2Ey_e + F = 0 \qquad (1)$$

Or, in quadratic form

$$(x_e y_e 1) \begin{bmatrix} A & B & D \\ B & C & E \\ D & E & F \end{bmatrix} \begin{pmatrix} x_e \\ y_e \\ 1 \end{pmatrix} = 0 \qquad (2)$$

RUNE-Tag Detection

Ellipse rotation estimation

A bundle of straight lines passing through the optical center and the ellipse defines an oblique elliptical cone

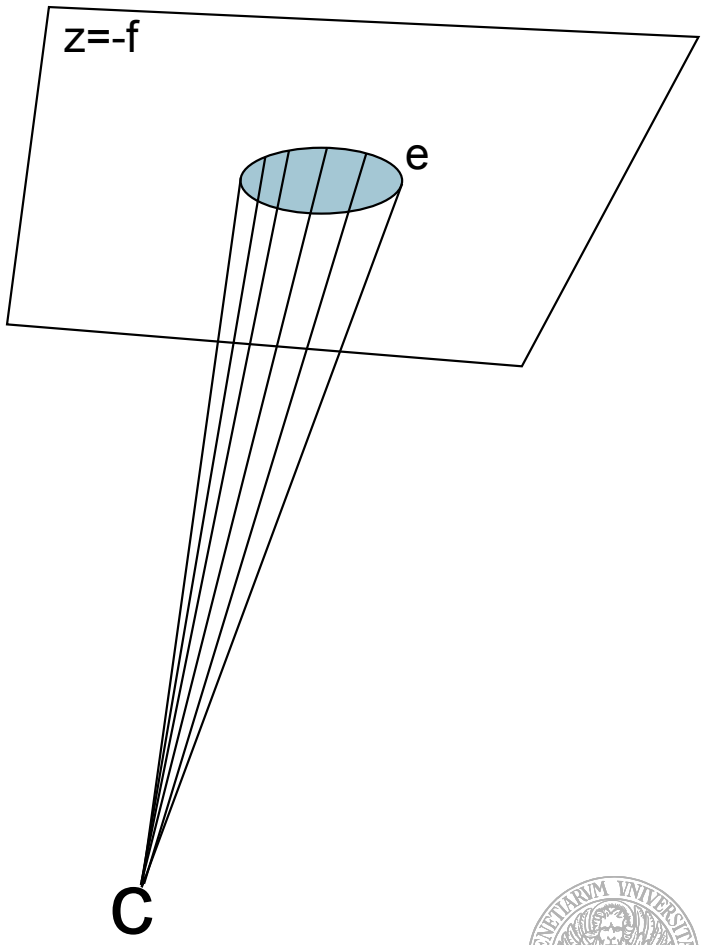
$$\mathbf{P} = k(x_e \; y_e \; -f)^T \qquad (3)$$

From (2) and (3), the equation that describe the elliptical cone is:

$$\mathbf{P^T Q P} = 0 \qquad (4)$$

Where:

$$\mathbf{Q} = \begin{bmatrix} A & B & -\frac{D}{f} \\ B & C & -\frac{E}{f} \\ -\frac{D}{f} & -\frac{E}{f} & -\frac{F}{f^2} \end{bmatrix} \qquad (5)$$



RUNE-Tag Detection

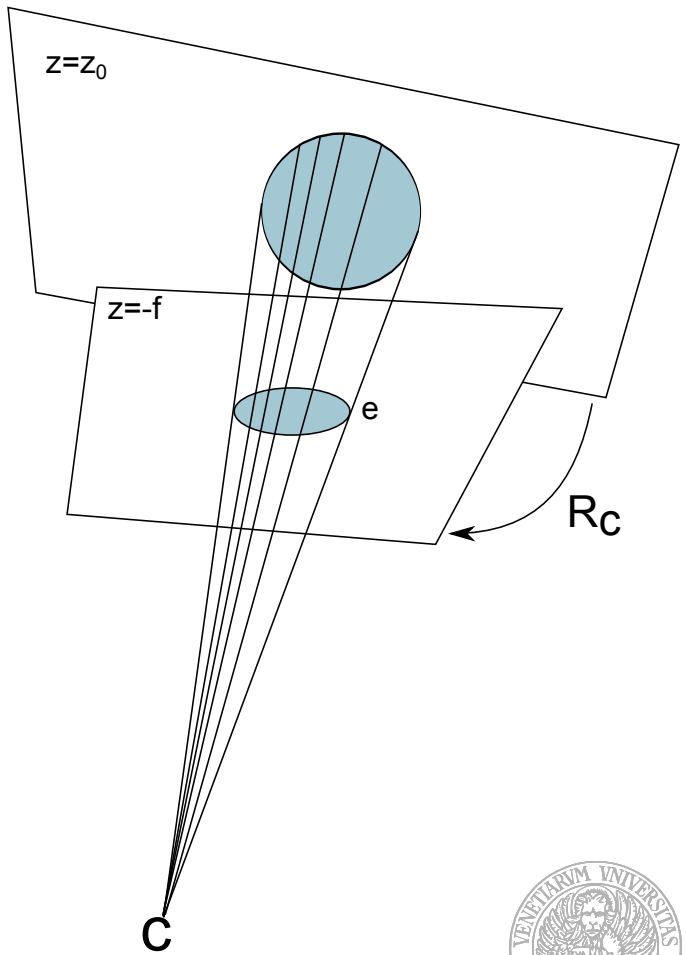
Ellipse rotation estimation

We consider a supporting-plane coordinate system whose origin is C but the Z-axis is defined by the normal vector of the supporting plane of the circle to be viewed.
A bundle of straight lines passing through the optical center and the circle defines an oblique circular cone:

$$\mathbf{P_c^T Q_c P_c} = 0 \qquad (6)$$

Where:

$$\mathbf{Q_c} = \begin{bmatrix} 1 & 0 & -\frac{x_0}{z_0} \\ 0 & 1 & -\frac{y_0}{z_0} \\ -\frac{x_0}{z_0} & -\frac{y_0}{z_0} & -\frac{x_0^2+y_0^2-r^2}{z_0^2} \end{bmatrix} \qquad (7)$$



RUNE-Tag Detection

Ellipse rotation estimation

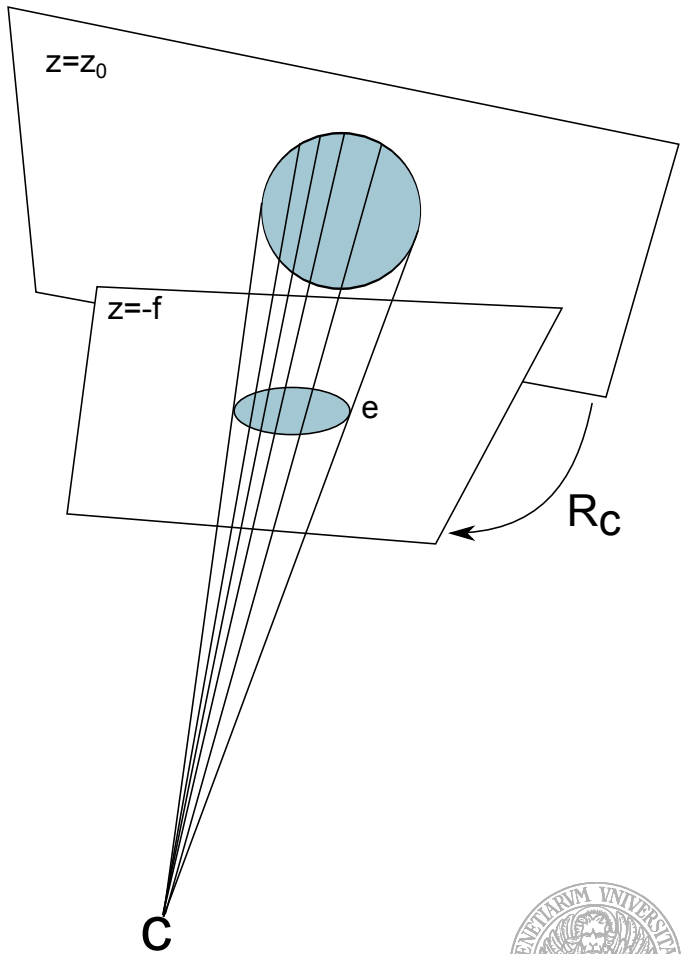
- The transform between the two coordinate systems is a rotation.
- Two cones share the same cone surface, so exits a rotation $\mathbf{R_c}$ such that:

$$\mathbf{P} = \mathbf{R_c}\mathbf{P_c} \tag{8}$$

Since $k\mathbf{Q_c}$ describe the same cone as $\mathbf{Q_c}$ $\forall k \neq 0$, from we have:

$$k\mathbf{R_c^T}\mathbf{Q}\mathbf{R_c} = \mathbf{Q_c} \tag{9}$$

Our goal: Find $\mathbf{R_c}$ and $\mathbf{Q_c}$



RUNE-Tag Detection

Ellipse rotation estimation

- We start by decomposing $\mathbf{Q}$ via SVD

$$\mathbf{Q} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \tag{10}$$

With $\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, \lambda_3)$.
By substituting (10) into (9) we got:

$$k\mathbf{R}^T\mathbf{\Lambda}\mathbf{R} = \mathbf{Q_c} \tag{11}$$

Where:

$$\mathbf{R} = \mathbf{V}^T\mathbf{R_c} \tag{12}$$

RUNE-Tag Detection

Ellipse rotation estimation

By simplifying equation (11) we obtain:

$$\mathbf{R} = \begin{bmatrix} g\cos\alpha & s_1 g\sin\alpha & s_2 h \\ \sin\alpha & -s_1\cos\alpha & 0 \\ s_1 s_2 h\cos\alpha & s_2 h\sin\alpha & -s_1 g \end{bmatrix} \tag{13}$$

Where:

$$g = \sqrt{\frac{\lambda_2 - \lambda_3}{\lambda_1 - \lambda_3}}, \quad h = \sqrt{\frac{\lambda_1 - \lambda_2}{\lambda_1 - \lambda_3}}$$

- α in an arbitrary rotation around the normal of the ellipse plane
- s_1 and s_2 are free signs.

RUNE-Tag Detection

Ellipse rotation estimation

By simplifying equation (11) we obtain:

$$\mathbf{R} = \begin{bmatrix} g cos\alpha & s_1 g sin\alpha & s_2 h \\ sin\alpha & -s_1 cos\alpha & 0 \\ s_1 s_2 h cos\alpha & s_2 h sin\alpha & -s_1 g \end{bmatrix} \quad (14)$$

- If we fix α we have 4 possible rotations depending by s_1 and s_2 values.
- If we require that $\langle \mathbf{VR}(0\ 0\ 1)^T, (0\ 0\ 1)^T \rangle > 0$ we obtain only two possible pairs (rotations).

RUNE-Tag Recognition

Coding Strategies

Once a possible tag is located a code is generated based on the presence or not of dots inside each slot.

We now need to tackle two coupled problems:

- Recognize the specific marker we are dealing with
- Find an alignment around the orthogonal axis on the marker, and match each detected dot with the model.

Mis-detections and occlusions make the matching non-exact!

RUNE-Tag Recognition

Coding Strategies

We decided to cast the problem into the well developed mathematical framework of coding theory.

- We can give guarantees about how many dots can be occluded without hinder the detection
- All dots have the same importance with respect to occlusions
- Great flexibility between number of tags that can be generated versus occlusion resilience

RUNE-Tag Recognition

Coding Strategies

- A **block code** of length n over a set of symbol S is a set $C \subset S^n$ and the element of the code are called **codewords**
- A **linear code** C is a k-dimensional sub-space of $(\mathbb{F}_q)^n$ where symbols are taken over the field $\mathbb{F}_q$
- A linear code C of length n and dimension k over the field $\mathbb{F}_q$ has q^k distinct codewords and is subject to singleton bound $dH(C) \leqslant n - k + 1$

In our setting we do not have a starting position of the code, so we have to take into account all cyclic shifts of a pattern.

RUNE-Tag Recognition

Coding Strategies

A linear code C is called **cyclic** if any cyclic shift of a codeword is still a codeword:

$$(c_0, \ldots, c_{n-1}) \in C \Rightarrow (c_{n-1}, c_0, \ldots, c_{n-2}) \in C$$

There is a bijection between the vectors of $(\mathbb{F}_q)^n$ and the residue class of the polynomial ring $\mathbb{F}_q[x]$ modulo division by $x^n - 1$

$$v = (v_0, \ldots, v_{n-1}) \Leftrightarrow v_0 + v_1 x + \cdots + v_{n-1} x^{n-1}$$

RUNE-Tag Recognition

Coding Strategies

Multiplying a polynomial from of a code by x modulo $x^n - 1$ corresponds to a cyclic shift:

$$x(c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}) = c_{n-1} + c_0 x + \cdots + c_{n-2} x^{n-1}$$

All cyclic codes in polynomial form are multiples of a **monic generator polynomial** $g(x)$ which divides $x^n - 1$ in $\mathbb{F}_q[x]$.

- The choice of the generator polynomial gives the tradeoff between the number of distinct codes and the number of misdetections that can be corrected

RUNE-Tag Recognition

Coding Strategies

Since all cyclic shifts are codes, we can group codewords into cyclic equivalence classes.

- The choice of the marker is encoded by the cyclic equivalence class
- The actual alignment of the circles can be obtained from the detected element within the class

In our first implementation, all possible shift of the detected marker is checked against all possible marker models and the one with the minimum hamming distance is returned.

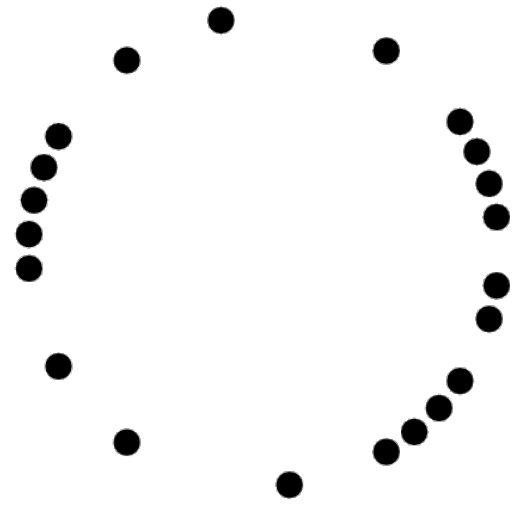
- We are working on direct decoding of the codewords

RUNE-Tag Recognition

RUNE-43

In the single-level Tag proposed the pattern is encoded as a vector in $(\mathbb{Z}_2)^{43}$ with the following generator:

$$g(x) = (1+x^2+x^4+x^7+x^{10}+x^{12}+x^{14})$$
$$(1+x+x^3+x^7+x^{11}+x^{13}+x^{14})$$



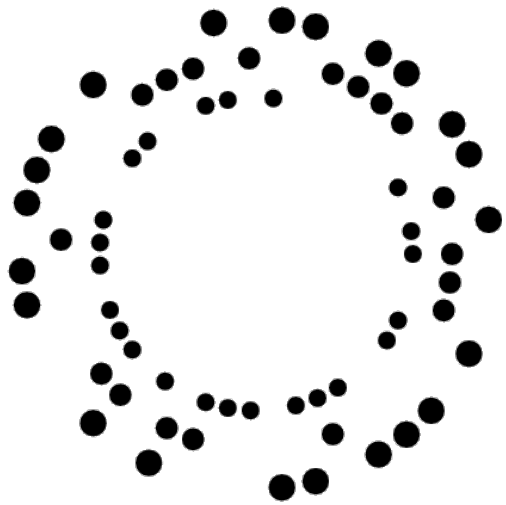
- Cyclic code of dimension 15
- 762 equivalence classes (markers)
- $dH(C) = 13$
- Correct up to 6 errors

RUNE-Tag Recognition

RUNE-129

In the 3-levels Tag proposed the pattern is encoded as a vector in $(\mathbb{Z}_7)^{43}$ with the following generator:

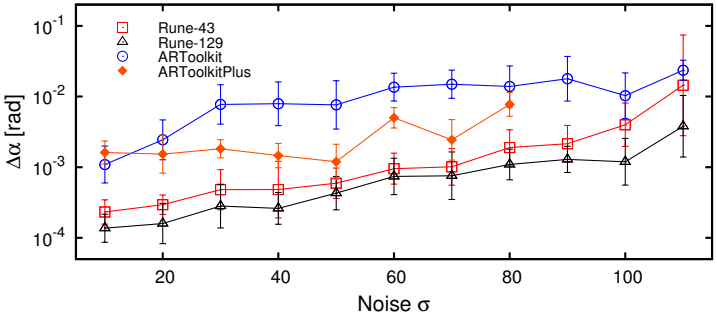
$$g(x) = (1 + 4x + x^2 + 6x^3 + x^4 + 4x^5 + x^6)$$
$$(1+2x^2+2x^3+2x^4+x^6)(1+x+3x^2+5x^3+3x^4+x^5+x^6)$$
$$(1+5x+5x^2+5x^4+5x^5+x^6)(1+6x+2x^3+6x^5+x^6)$$
$$(1 + 6x + 4x^2 + 3x^3 + 4x^4 + 6x^5 + x^6)$$

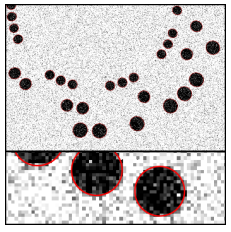


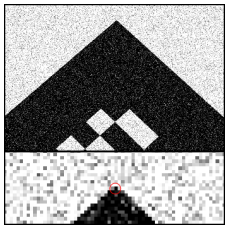
- Cyclic code of dimension 7
- 19152 equivalence classes (markers)
- $dH(C) = 30$
- Correct up to 14 errors or 29 erasures

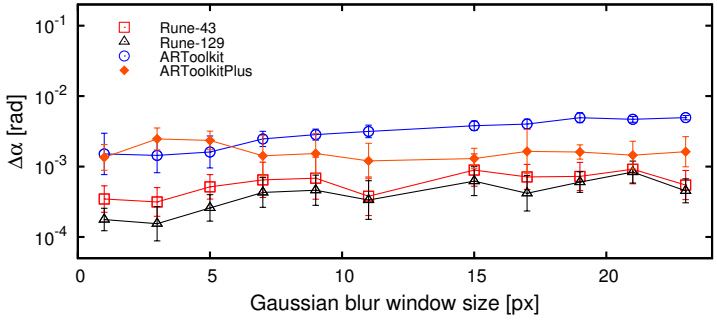
RUNE-Tag

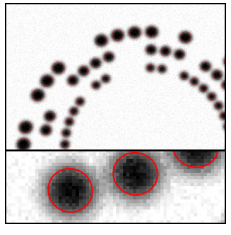
Experimental evaluation

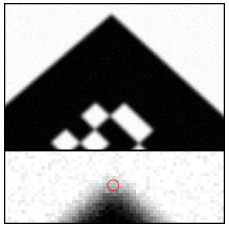


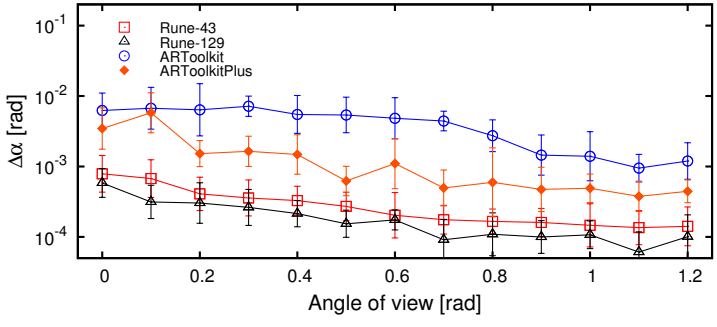


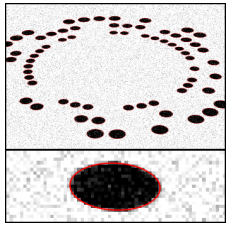


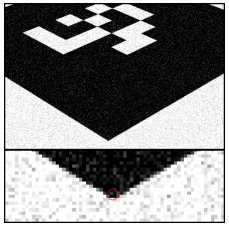






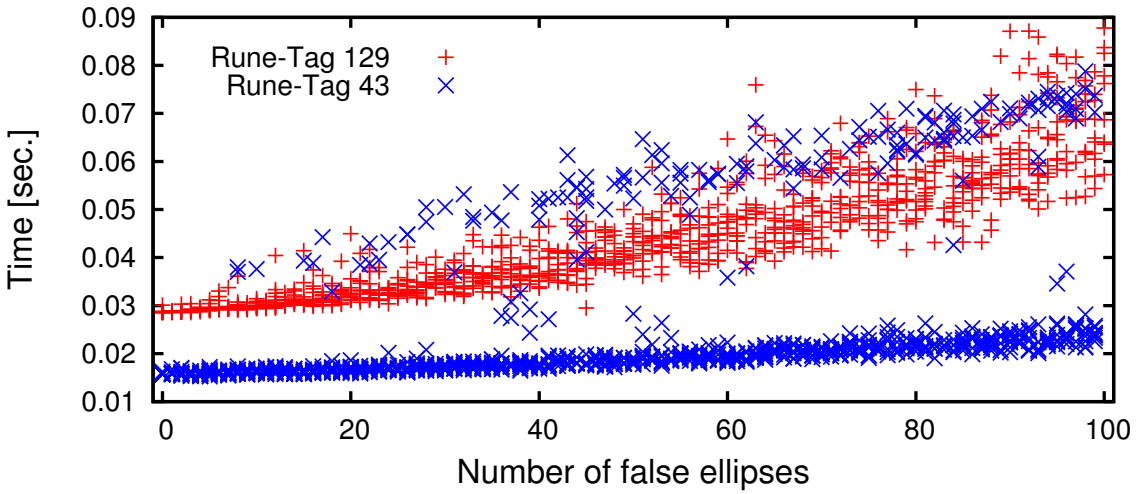


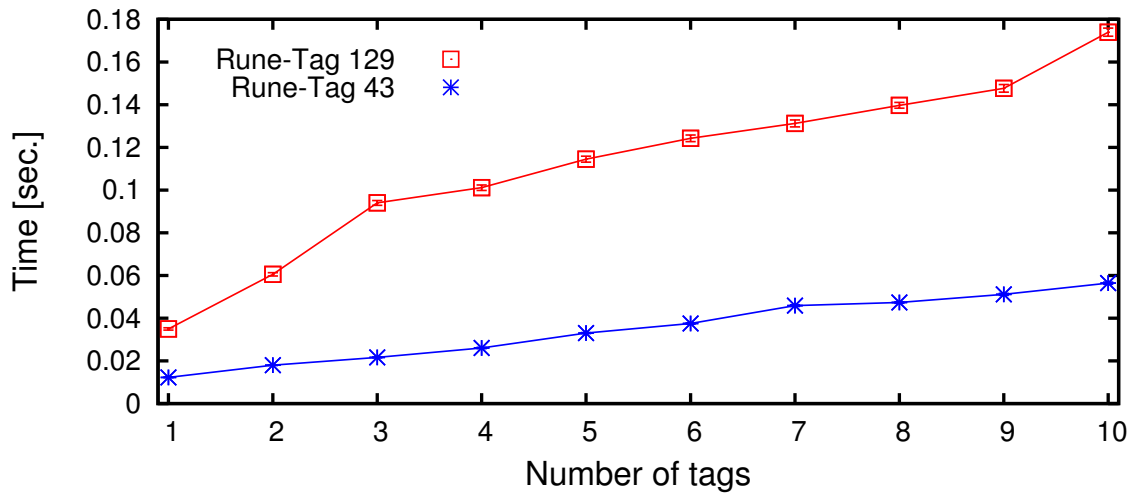




RUNE-Tag

Experimental evaluation





Conclusions

Main features

We proposed a novel type of fiducial markers with some new interesting features:

- RUNE-Tags exceed the current state-of-the-art about the accuracy of the computed pose under various limiting conditions
- Very robust against severe occlusions
- The interior of the tag is free for additional payload
- Tunable trade-off between occlusion resilience and the number of possible tags
- While slower than AR specific tags, they are still fast enough for real-time augmented reality on today's PC hardware

Conclusions

Limitations and future works

Some limitations:

- Severe packing of dots can lead the ellipse detection to merge features at low resolution
- Camera calibration required to detect the tags

Future works:

- Direct cyclic codes decoding
- Inclusion into OpenCV library
- Tag detection without camera calibration

Thank you.
Any question?

References

Qian Chen, Haiyuan Wu, and Toshikazu Wada.

Camera calibration with two arbitrary coplanar circles.
In *European Conference on Computer Vision - ECCV*, 2004.

Mark Fiala.

Designing highly reliable fiducial markers.
IEEE Trans. Pattern Anal. Mach. Intel., 32(7), 2010.

L. Gatrell, W. Hoff, and C. Sklair.

Robust image features: Concentric contrasting circles and their image extraction.
In *Proc. of Cooperative Intelligent Robotics in Space*, Washington, USA, 1991. SPIE.

William Grimson.

Object recognition by computer - the role of geometric constraints.
MIT Press, 1990.

Hirokazu Kato and Mark Billinghurst.

Marker tracking and hmd calibration for a video-based augmented reality conferencing system.
In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality*, Washington, DC, USA, 1999. IEEE Computer Society.

Leonid Naimark and Eric Foxlin.

Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker.
In *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, ISMAR '02, Washington, DC, USA, 2002. IEEE Computer Society.

Jean Ouellet and Patrick Hebert.

Precise ellipse estimation without contour point extraction.
Mach. Vision Appl., 21, 2009.