Dependable Coding of Fiducial Tags

Andrew C. Rice
Laboratory For Communication Engineering
University of Cambridge
acr31@cam.ac.uk

Christopher B. Cain
Department of Pure Mathematics and Mathematical Statistics
University of Cambridge
c.b.cain@dpmms.cam.ac.uk

John K. Fawcett
Laboratory For Communication Engineering
University of Cambridge
jkf21@cam.ac.uk

19th September 2004

Abstract

Fiducial tags can be recognized successfully and decoded by computer vision systems in order to produce location information. We term a system dependable if its observable results are predictable and repeatable. The dependability of such a vision system is fundamentally dependent on the scheme used to encode data on the tag. We show that the rotational symmetry common to many tag designs requires particular consideration in order to understand the performance of the coding schemes when errors occur. We develop an abstract representation of tags carrying symbolic data which allows existing information coding techniques to achieve robust codes. An error-correcting coding scheme is presented for carrying arbitrary symbolic data in a dependable vision system.

1 Introduction

High precision location information is an excellent source of context for many ubiquitous applications. Unfortunately, the sensing systems required to source such information are expensive to build, deploy and maintain—GPS is an example of this. Location systems that can derive information from commodity components such as WaveLAN or Bluetooth understandably fall short of the level of precision provided by more specialized, expensive systems.

Vision systems are an exception. They can be constructed from commodity components and yet have the potential to provide highly accurate information (c.f. photogrammetry). Modern cameras and CCD arrays can provide good quality images[1], whilst distortion caused by lensing systems[2] can be corrected to accuracies better than 0.01 pixels[3]. Fiducial tags are often used as markers to simplify the image recognition process. They present particular features to allow the application of faster processing algorithms and more robust recognition than for unconstrained vision systems.

Recently, researchers have begun to acknowledge the importance of reliability for ubiquitous computing[4]. The new users of context-aware systems will not have technical backgrounds or in-depth understanding of sensing systems and yet must be convinced to trust these systems to assist in their tasks. The system must display predictable behaviour upon which users will base their mental models of the system. A system is considered *dependable* its observable results are predictable and repeatable.

In order to ensure that a location system is as dependable as possible we must ensure that every algorithm and process used within the system is robust and that the errors inherent within the system are understood. Identity is as a primary source of context useful to ubiquitous computing[5]. The robustness of identity information for vision systems relates to the mechanism used for encoding information identifier on the fiducial tag.

Squares and circles are common choices of fiducial shape for vision systems. Each of these shapes displays rotational symmetry that reduces the expected difference between each unique payload. We present a classification of coding systems for black and white fiducial tags and suggest a suitable design separation between tags and coding schemes so each can be considered in isolation. In particular, the importance of cyclic codes is demonstrated when considering the rotational symmetry of planar tags. Finally, we present new, robust

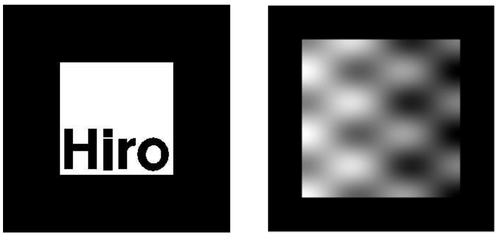

Figure 1: A sample template from the ARToolKit distribution (left) and from the maximum distance set (right).

coding schemes for encoding symbolic data as the tag payload.

2 Template-Based Codes

A template code encodes an identifier as a pattern that is decoded by searching a database of possible patterns for the closest match. ARToolKit[6] uses a set of manually-chosen patterns encoded onto a square tag. The imaged tag is compared, after perspective correction, using an auto-correlation co-efficient with a database of all issued patterns. The four-fold rotational symmetry of the tag is accommodated by comparing the template in all four corresponding rotations. The designers encourage users to select tag designs with strong asymmetric features. The purported advantage of this method is that the tag designs can be selected to have semantic meaning for the users of the system as any image can be used. However, the ad hoc selection for tag templates means that the system cannot guarantee good separation of the targets.

Owen *et al.* present a scheme for selecting a set of greyscale (asymmetric) templates with maximum autocorrelation distance[7]. This process creates approximately 200 maximally separated tags, but the tags no longer have semantic meaning for a human reader. Figure 1 shows an example template from ARToolKit and one using the maximum separation scheme.

Template schemes present a number of problems for a dependable vision system. Firstly, an analysis of the set of templates must be performed to verify that the autocorrelation between any pair of patterns is small and so incremental deployment is difficult. Secondly, perspective projection combined with the limited resolution of CCD cameras will introduce distortion into the imaged template affecting the correlation co-efficient; current schemes for template selection do not take this into account. Thirdly, noise produced by the image acquisition phase (e.g. dark current) will introduce additional distortion to the templates. Finally, wide-scale deployment of a vision system will require a large address space that cannot be provided with this mechanism. For example, in our lab of approximately 40 people we have 206 tagged items—already exceeding the limit of the maximum distance method.

3 Symbolic Codes

A more promising approach for coding data on tags is a symbolic method. The tag is divided up into data cells, each of which is capable of storing a symbol. This approach presents a number of potential advantages over template-based codes: we expect to be able to achieve a substantially larger address space and error detecting or error correcting codes give the ability to detect or recover from image noise. If human interpretation of the tags is required a tag design could easily include human readable text or icons in addition to the machine readable coding.

Readers of symbolic tags should expect both single bit and burst errors. A bit error could occur due to the image being sampled at incorrect points or due to noise from the CCD array. Partial occlusions of the tags or complex lighting conditions will cause burst errors and an entire sequence of symbols will be misread. We assume that these errors are equally likely to occur across the whole tag.

The current generation of symbolic tags does not take full advantage of the error handling potential of symbolic codes due to the rotational symmetry of the tags. For example, TRIP[8] uses circular tags with two rings of data split into sectors. Each sector stores one of four possible symbols (each ring within the sector stores a binary value). The symbol corresponding to a completely black sector is reserved for a synchronization sector. The remaining tag consists of two checksum sectors and the payload encoded as a base 3 number (because the fourth symbol is reserved for synchronization). Despite the (weak) error detection properties of the checksum the code is limited by the unprotected synchronization sector. As a result this scheme can only ever guarantee to detect one bit of error; two bits of error suffice to fool the system into starting decoding from the wrong sector. Whether or not this invalid reading will pass the checksum depends on the data that was encoded.

The Matrix Tag system[9] uses square tags to carry arbitrary payloads with CRCs appended on the ends. This approach lacks robustness because the tag has four-fold rotational symmetry. Thus, rotated tags read as permutations of the original code. We have no analytical way to determine whether or not these permutations will contain a valid CRC.

Zhong *et al.* present a square tag which carries 5 bits of data protected by a block sum code check or 6 bits of data protected by a Hamming code[10]. The four corner bits are used for orientation to ensure that the correct code can be read from the tag. Unfortunately, the block code does not protect these orientation bits and so two bits of error in the image can result in the system reading a rotated tag from the wrong orientation and thus returning an invalid code. The Hamming distance of this code should thus be considered to be only two bits until it can be proven that no two codes are rotationally self-similar.

Cho and Neumann encode data on their multi-ring circular tags using *solid* rings chosen from n colours[11]. Assuming these colours can be reliably identified by the system, the method has the potential to be robust because the code can be read radially at any position. However, the amount of data that can be stored on the tag is small due to the large amount of redundancy. Also, an additional error correcting code would be needed if error correction capability was required.

4 Rotational Invariance

The rotational symmetry of tags means that permutations of the codes can be read. Some current code designs attempt to resolve this problem by introducing anchor points in to the code (such as TRIP's synchronization sectors or Zhong's orientation bits) but fail to protect these data when protecting the payload. Other systems have relied on the error detection capabilities of the coding scheme to additionally detect rotations of the tag data. The chance of collisions due to this approach cannot be analyzed with existing information coding theory.

Cyclic codes present a solution to the rotational symmetry problem. One property of a cyclic code is that any rotation of a valid codeword is also a valid codeword. If we arrange data coded with a cyclic code in such a way that rotations of the tag correspond to rotations of the sampled data (rather than general permutations) then we can be assured that the error detecting or error correcting capabilities of the code will be unaffected. This separates the code from tag design details and enables a mathematical analysis of code capability. To see that this is true consider the following scenario. Suppose the minimum distance of the original code is d and given a valid codeword we introduce an error in less than d places. If the resulting word is equal to the rotation of some codeword then it itself must be a codeword (all rotations of codewords are codewords!). This contradicts the fact that the original code had minimum distance d.

We shall use the term *rotational invariance* for reading a code from a tag such that all symmetric rotations of the tag correspond to rotations of the code. It is straightforward to arrange to read a circular tag in a rotationally invariant manner (Figure 2). Arranging to read a square tag in this manner is less geometrically intuitive. For example, the immediately-appealing raster approach produces a different permutation of the code depending on the starting corner. A scheme that reads the tag as four triangular sections achieves rotational invariance (Figure 3) but if the code grid contains an odd number of data cells then the central cell cannot be used.

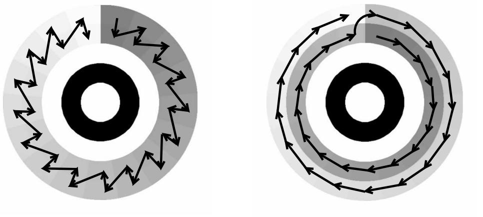

Figure 2: A circular tag read can be read in a rotationally invariant manner (left) or a non-invariant manner (right).

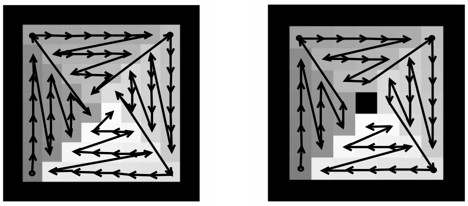

Figure 3: A square tag can be read in a rotationally invariant manner (each reading could start in any of the four corners). Tags with an odd number of cells must sacrifice the central cell.

If we apply a cyclic code to a tag in a rotationally invariant manner we know that the error detecting or error correcting properties of the code will not be affected by the rotational symmetry of the tag. However, this presents an additional problem because the system will be unable to select the correct code from the set of possibilities read from the tag. Each possibility will appear as a valid codeword (after applying any applicable error correction routine). One approach is to select the particular rotation which has a smaller value than every other possibility. This means that for each value coded onto a tag there will be a number of additional codewords which also decode to the same value. We call codes exhibiting this property *symbolic identifier* codes—the code cannot store arbitrary data (without using a non-systematic code).

We can characterize a tag's data-carrying capability in terms of two variables: **Symbol Size** is the number of bits allocated to storing each symbol. If the tag is rotated by one place and the code re-sampled, the new value should be identical to the previous value after a rotation through *symbol size* bits; **Payload Size** is number of symbols the tag can store.

A circular tag with m rings and n sectors thus has a symbol size of m bits and a payload size n. A $2p \times 2p$ square tag laid out using the rotational invariant scheme in Figure 3 has symbol size of p^2 bits and payload size 4 whereas a $(2p+1) \times (2p+1)$ tag has a symbol size of $p(p+1)$ bits and payload size of 4 symbols.

We can also parameterize coding schemes in a similar way. The number of symbols corresponds to the size of the field used to define the polynomials in the cyclic code. For example, the various generator polynomials for a CRC are defined over the field with two elements (symbol size is 1-bit). Reed Solomon codes (a subset of BCH codes), which are used for error correction on CDs and DVDs (among other things), can be defined for fields of size 256 (symbol size is 8 bits). Of course, if the tag provides a symbol set of size 8 then codes requiring a symbol size less than 8 can be accommodated by packing additional symbols into each sector (with a corresponding increase in the payload size).

The payload size must equal the block length of the cyclic code. This precludes the use of CRCs: the generator polynomial for CRC-CCITT (a 16 bit CRC) has a block length 32767 bits. Typically, a CRC is used with much smaller messages than this—the unused bits are assumed to be zero and not transmitted. For a symmetric tag we do not have this luxury because we must transmit the zeros as well in order for all rotations of the code to be valid codewords. A circular tag carrying CRC-CCITT data would need 151 rings and 217 sectors!

5 Robust Data Coding Schemes

We have identified the concept of rotational invariance, that allows robust application of cyclic codes to symmetric tags. This exposes the ambiguity introduced into the coding system and hence reduces the codes to carrying only identifiers which can then be used for a database lookup, rather than symbolic data. Furthermore, we have shown that existing techniques that use synchronization sectors or orientation bits to anchor the code are not capable of coping with more than 1 bit of error. It is possible to design synchronization sectors that can withstand more than one bit of error at the expense of a reduced payload. We now present some additional coding schemes that allow tags to carry arbitrary data robustly.

5.1 Simple Parity Code

A bit string with parity at the end fulfills our criteria for a rotationally invariant code: every rotation of the coded data should also have valid parity. To generate a code we take a tag with payload size p and symbol size s and encode $s(p-1)$ bits and an additional parity symbol. This is an example of a code that can only encode an identifier because the decoded message must be rotated round until the minimal value is found. However, it achieves the same minimum hamming distance as the TRIP code and the Hamming code scheme by Zhong *et al.* and can store considerably more data.

5.2 Independent Chunk Code

Given a tag with a large *symbol size*, each symbol is considered as a separate codeword which is protected by an error detecting or error correcting code. The first bit of each symbol is used to anchor the code: the first bit of the first symbol is set and the first bit of every other symbol is unset. For example, a square tag of size 8×8 has a symbol size of 16 and a payload size of 4 symbols. We can encode a 44-bit payload in four 11-bit chunks. Each symbol on the tag contains one chunk, one orientation bit, and a 4-bit CRC (Figure 4). This code is at least as strong as the 4-bit CRC used for each symbol, if the designer required stronger error detection or error correction then a different code can be used for each symbol. In the cases where errors occur evenly over the tag rather than concentrated in one sector this code should be rather stronger than a single CRC-4. The orientation bits are included in the CRC-4 for additional reliability. The drawbacks of using this code is that 4 bits of every symbol are used to get the same Hamming distance as traditional use of a single 4-bit CRC. Additionally a further bit is required per symbol to orient the code. The advantage of this encoding method is that the code need not have rotational invariance and so a truncated CRC is permissible.

5.3 Structured Cyclic Code

Our third scheme is a more conventional cyclic code with additional structure that encodes the amount of rotation that the code has undergone. The full details are presented in Appendix A. A generator polynomial f is chosen that will produce a code with the desired error detecting or error correcting capabilities. The target tag has a symbol size s and a payload size n. An auxiliary generator polynomial h, dependent on f, is then found and a primitive polynomial ω is found from h. These parameters are fixed for a particular instance of this coding

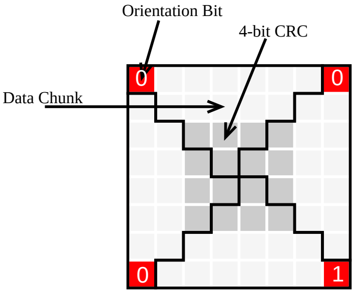

Figure 4: The Independent Chunk Code operates on a tag with large symbols, each of which contains an orientation bit and some error protection.

scheme and so the computational costs of finding them is not a run-time issue.

We encode a message m of $n - \deg(f) - \deg(h)$ symbols and an arbitrary number α, $0 \leq \alpha < (2^{s\deg(h)} - 1)/n$ by careful choice of a check polynomial c based on ω,α,m,h and f. The data (α, m) are encoded as $X^r m + c$ i.e. the message m is left-shifted by r symbols and the check polynomial c is written into the low bits. This will be a valid codeword for the generator polynomial f and so traditional error correction routines from the literature[12] can be applied. The additional structure imposed on our check polynomial c further provides a means of recovering the amount of rotation the code has undergone (in addition to α).

6 Evaluation

We used a test system to evaluate the performance of our new cyclic coding schemes. A circular tag with 5 rings and 31 sectors was used to carry a payload encoded with each of the new schemes.

- **TRIP** Adaption of the original coding technique used in the TRIP system: 1 synchronization sector followed by 2 checksum sectors and 28 payload sectors encoded base 31.
- **SPC** Simple Parity Code: 154 payload cells (not sectors) followed by 1 parity cell encoded base 2.
- **ICC** Independent Chunk Code: 31 independent chunks (one per symbol) containing 1 orientation bit, 1 parity bit and 3 bits of payload;
- **SCC-1** Structured Cyclic Code with f chosen as in a Reed-Solomon code giving 3 symbols of separation between codewords.
- **SCC-2** Structured Cyclic Code with f chosen as in a Reed-Solomon code giving 11 symbols of separation between codewords.

Name	Message Length (bits)	Hamming Distance
TRIP	139	2 symbols
SPC	154	2 bits
ICC	93	2 bits per symbol
SCC-1	141	3 symbols
SCC-2	101	11 symbols

Figure 5: The data carrying capabilities of the evaluated coding schemes. 155 data cells are available on the tag.

The data carrying capabilities of each of these codes are given in Figure 5.

An OpenGL test harness was used to render tags fully facing the camera at a distance 2 times the tag width and 1000 trials per code with each coding scheme were run. Gaussian noise (mean 0 and standard deviation 53) was injected into the images and the target tags decoded. We define three possible results from each test run.

- **Successful Read** The payload on the tag is decoded and the returned code matches the value encoded (a true positive).
- **Failed Read** The payload on the tag fails to decode and so the system fails to recognize a tag.
- **False Read** The payload on the tag is decoded but the returned code does not match the encoded value, i.e. the error detection built into the code is defeated (a false positive).

Figure 6 shows the percentage of frames for each code that contained successful readings, failed readings and false readings. Normalized values for these percentages are obtained by multiplying by the proportion of the utilized address space. The results confirm that allocating more bits to error control strengthens the code. The SCC-2 shows a particularly high successful read rate due to its large error-correcting capability. This redundancy also gives it a false read rate small enough that it failed to manifest itself in our 1000 samples. The error correction ability of the SCC-1 code increases the successful read rate above that of the non-correcting codes at the expense of increasing the false read rate. The TRIP, SPC, and ICC codes have the same minimum hamming distance. However, the noise was evenly distributed across the whole image and so the ICC code's parity bits acted mostly independently giving it good false read rate. The TRIP code distributes the code particularly unevenly over the tag, this manifests itself in the code's more variable behaviour than the ICC code—it shows an increased successful read rate *and* an increased false read rate even though there is no attempted error correction. For interactive systems, designers might choose to minimize the false read rate at the expense of a higher failed read rate because users can be expected to retry a tag if it fails to read.

For a circular tag we wish to sample data points at the centre of each sector. To achieve this we need to apply an offset to the angle of each sector. However, the symmetry of a circular tag means that this offset is unknown. One scheme for achieving a reliable reading is to attempt to read the code n times from the tag at intervals of $1/n$ sectors. The system can then select the correct reading by looking for duplicate readings. From a viewpoint of coding robustness this acts similarly to a repetition code: the same bit error must be present in two or more of the readings in order for it to be considered as a valid code. In practice this reduces the false error rate almost to zero for each of the coding techniques. Of course, if the errors in the decoding are systematic—perhaps due to an occlusion or lighting conditions—this will not be as successful. Also, use of this method reduces the success rate of the code as well, especially for the error correcting codes.

A further experiment using the TRIP, SPC, and ICC tests was performed using a square tag of size 12×12 rather than a circular tag. The SCC codes cannot be applied to square tags due to the constraints imposed on the message size. The ICC code for a square (Figure 7) is rather more efficient than for a circular tag due to the increased symbol size. We expect the square tag to perform less well as compared to the circular tag because its payload area is smaller. Figure 8 shows the various decoding rates which bear out the same trends as for the circular tag. This provides some justification for our argument that code selection can be done in isolation from the actual tag design. The ICC(square) code presents a better normalized successful read rate than the TRIP(square) code which is contrary to the results for circular tags. This is because the ICC code is much more efficient for tags with large symbol sizes and so its success rate is boosted to acknowledge this. However, the increased symbol size means that there will be fewer parity bits embedded in the code—this is reflected by the increased false read rate for ICC(square) over ICC(circle).

7 Asymmetric Tags

Another approach is to introduce an asymmetric feature into the tag design thus permitting use of conventional coding systems. For example, Foxlin[13] presents a system that uses an off-centre eyelet for this purpose. This addition to the tag design should be as unintrusive as possible as it will reduce the size of the data carrying area of the tag. However, we must also ensure that the noise in the image does not cause us to mis-read the orientation of the tag.

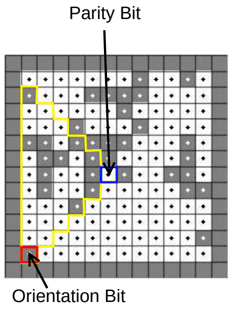

Figure 7: A square tag carrying data with the ICC coding scheme. The tag (faded to grey) has been sampled at the points shown. One symbol, including its orientation bit and parity bit, has been highlighted.

In order to maximise the data carrying capability of the tag we require that the new feature and the data carrying code are equally resistant to image noise—correct choice of the tag orientation is useless if the code cannot be subsequently decoded. It is difficult to quantify the strength of the introduced asymmetric feature and so designers must err on the side of safety.

QR Codes are a popular 2-dimensional barcode that use a particular pattern on three corners to orient the tag. Four different levels of error correction are available of which level ‘M’ corresponds most closely to the level afforded by the SCC-2 code presented above. QR Codes are available in a number of sizes, the largest of which has a data area with dimensions 177×177. This size of tag can store 18648[1] bits which corresponds to a utilization of 59.5% whereas the SCC-2 code has a utilization of 64%. The primary reason for this is that the area occupied by the asymmetric features added to the QR Code is disproportionately large compared with the error correction capability of the error correcting code.

Use of symmetric tags and rotationally invariant codes is advantagous in this respect because the minimum amount of payload space is wasted in order to encode rotation information. Also, rotationally invariant tags result in the least possible complication of the computer vision aspect of the decoding. The system need only read the data from the tag rather than search for additional features before decoding the information.

8 Conclusion

A dependable location system requires robust, predictable behaviour for every element of its operation. The choice of coding scheme used to store data on a

[1] See http://www.denso-wave.com/qrcode/vertable4-e.html

Name	Successful Read		Failed Read		False Read	
	%	Normalized	%	Normalized	%	Normalized
TRIP	24.0	21.5	74.0	66.4	2.00	1.79
SPC	31.0	30.8	46.0	45.7	23.0	22.9
ICC	27.1	16.3	72.6	43.6	0.300	0.18
SCC-1	55.7	50.7	6.20	5.64	38.1	34.7
SCC-2	94.0	61.3	6.00	3.91	$< \frac{1}{10}$	0

Figure 6: The success, failed, and false reading rates for a circular tag with 5 rings and 31 sectors.

Name	Successful Read		Failed Read		False Read	
	%	Normalized	%	Normalized	%	Normalized
TRIP(square)	12.6	9.5475	83.4	63.12917	4	3.027778
SPC(square)	27.4	27.4083	44	43.69444	28.6	28.40139
ICC(square)	24	22.6667	70.5	66.58333	5.5	5.194444

Figure 8: The successful, failed, and false reading rates for the evaluated coding schemes on a square tag of size 12×12.

fiducial tag is an important aspect often overlooked. Template-based systems pose a particular risk because it is difficult to analyse the effects of image noise or perspective projection on the pattern. Current symbolic coding schemes have not fully appreciated the rotational ambiguity caused by symmetric tags and thus do not have quantifiable characteristics. Sentient Computing must be dependable in order to fulfill its potential. This requires rigorously understood coding schemes and this work provides useful, successful, and efficient examples.

Tag abstraction coupled with the principle of rotational invariance allows tags to utilize existing information coding techniques and makes the result amenable to rigorous mathematical analysis. This decomposition highlights a particular class of symbolic codes capable only of encoding identifiers due to the rotational ambiguity of the tags. For systems requiring arbitrary symbolic data we have presented a number of dependable coding schemes.

When selecting a suitable code for a tag design we may choose to optimize based on the message size, error handling capabilities, implementation difficulty or computational cost. The Structured Cyclic Coding scheme presented herein proves to be a good choice: it makes efficient use of the available payload space to carry a large message whilst still providing a large degree of error correction. It allows one to encode arbitrary data and requires the minimal amount of computer vision possible. Future work will be to document the extension this technique to apply to square tags as well as circles.

Another approach for dealing with rotational symmetry was to introduce asymmetric features to the tag to break the ambiguity. However, designing the additional features required less efficient that the approach suggested here due to the (unmeasurable) trade-off between feature size and tag capacity.

A Robust Data Coding Scheme for Symmetric Tags

We operate in a Galois Field of size 2^s because this lends itself to an efficient implementation. Take a generator polynomial f for some cyclic code that has the desired error control properties. As in a normal cyclic code of length n we have $X^n + 1 = fg$ for some g and all valid codewords are multiples of f. We then find h which is an irreducible factor of g, and such that X has order n modulo h. Since h is irreducible, there exists a primitive polynomial ω whereby every non-zero value modulo h can be expressed as $\omega^k \mod h$ for some k. The order of ω is $l = 2^{s \deg(h)} - 1$. Since X has order n modulo h, it is possible to choose ω such that $\omega^{\frac{l}{n}} \equiv X \mod h$.

Let $r = \deg(f) + \deg(h)$. We can now encode a number $\alpha < \frac{l}{n}$ and a polynomial m where $\deg(m) < n - r$ as follows. Define the check polynomial c where $\deg(c) < r$ by the conditions $c \equiv \omega^\alpha + X^r m \mod h$ and $c \equiv X^r m \mod f$ (this can be done using the Chinese Remainder Theorem—see page 29 of [12]). We encode (α, m) as $c + X^r m$. This is a valid codeword since $c + X^r m \equiv X^r m + X^r m \equiv 0 \mod f$.

The received codeword y, read from the tag, should be a valid codeword for the generator polynomial f. If not, existing techniques for error correction[12] can now be applied to find the nearest valid codeword. If a codeword represented by the polynomial y is rotated one place round then the polynomial corresponding to the new codeword is given by $Xy \mod 1 + X^n$. Proof: if y is given by $a_0 + a_1 X + \ldots + a_{n-1}X^{n-1}$ then $Xy = a_0 X + a_1 X^2 + \ldots + a_{n-1}X^n$ and the rotated codeword is represented by $a_{n-1} + a_0 X + a_1 X^2 + \ldots + a_{n-2}X^{n-1}$. We see that the difference of these last two expressions is $a_{n-1}(1 + X^n)$, a multiple of $1 + X^n$.

Thus, if the (possibly corrected) codeword y read from the tag has been rotated by t places from the intended codeword then we will have $y = X^t(X^r m + c) \mod (X^n + 1)$.

We now show how to find t (and α). This can then be used to recover $X^r m + c$ and hence the stored value. Since h divides $X^n + 1$ we see that:

$$\begin{aligned} y &\equiv X^t(c + X^r m) \mod h \\ &\equiv X^t(\omega^\alpha + X^r m + X^r m) \mod h \\ &\equiv X^t \omega^\alpha \mod h \equiv \omega^{\alpha + t\frac{l}{n}} \mod h \end{aligned}$$

Thus, if we find the unique z such that $0 \leq z < l$ and $y \equiv \omega^z \mod h$ then:

$$\alpha = z \mod \frac{l}{n}, t = \lfloor \frac{z}{l/n} \rfloor \qquad (1)$$

Our original message m is the most significant $n - r$ coefficients of y after rotating it right by t places (dividing by X^t).

The task of finding z (solving a discrete logarithm) is potentially computationally expensive. However, in practice h is typically small enough to allow this to be solved using a lookup table. In particular, for Reed-Solomon codes $\deg(h) = 1$ and so l is very small. Even in less favourable circumstances l tends to have a large number of small factors and so even for large values of l (which bounds z), the discrete log can be found efficiently using the Pohlig-Hellman algorithm[14].

B Acknowledgements

This work was supported by the EPSRC. Thanks to Alastair Beresford and Andy Hopper for their support.

References

[1] M. R. Shortis and H. A. Beyer. *Close Range Photogrammetry and Machine Vision*, chapter 5, pages 106–155. Whittles Publishing, 1996.

[2] D.C. Brown. Decentering distortion of lenses. *Photogrammetric Engineering and Remote Sensing*, 32(3):444–462, 1966.

[3] Janne Heikklä and Olli Silvén. A four-step camera calibration procedure with implicit image correction. In *Computer Vision and Pattern Recognition*, pages 1106–1113, 1997.

[4] Jeffrey Hightower and Gaetano Borriello. Real-time error in location modeling for ubiquitous computing. In *Location Modeling for Ubiquitous Computing—Ubicomp 2001 Workshop*, pages 21–27, September 2001.

[5] Anind K. Dey and Gregory D.Abowd. Towards a better understanding of context and context-awareness. In *Proceedings of the CHI 2000 Workshop on "The What, Who, Where, When, Why and How of Context-Awareness"*, 2000.

[6] Mark Billinghurst and Hirokazu Kato. Collaborative mixed reality. In *Proceedings of the First International Symposium on Mixed Reality*, pages 261–284, 1999.

[7] Charles B. Owen, Fan Xiao, and Paul Middlin. What is the best fiducial? In *The First IEEE International Augmented Reality Toolkit Workshop*, pages 98–105, September 2002.

[8] Diego Loṕez de Ipinã. A low-cost vision-based location system for ubiquitous computing. *Personal and Ubiquitous Computing Journal*, 6(3):206–219, May 2002.

[9] Jun Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. In *Proceedings of Asia Pacific Computer Human Interaction*, pages 63–68, July 1998.

[10] Xiaowei Zhong, Peiran Liu, Nicolas D. Georganas, and Pierre Boulanger. Designing a vision-based collaborative augmented reality application for industrial training. *it-Information Technology*, 45(1):7–18, 2003.

[11] Youngkwan Cho and Ulrich Neumann. Multiring fiducial systems for scalable fiducial-tracking augmented reality. *PRESENCE: Teleoperators and Virtual Environments*, 10(6):599–612, December 2001.

[12] Elwyn R. Berlekamp. *Algebraic Coding Theory*. McGraw-Hill, 1968.

[13] Leonid Naimark and Eric Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *IEEE International Symposium on Mixed and Augmented Reality*, pages 27–36, September 2002.

[14] S. Pholig and M. Hellman. An improved algorithm for computing logarithms over gf(p) and its cryptographic significance. *IEEE Transactions on Information Theory*, 24(1):106–110, January 1978.