Error-Tolerant Cyclic Sequences for Vision-Based Absolute Encoders

Kevin C. Wolfe and Gregory S. Chirikjian

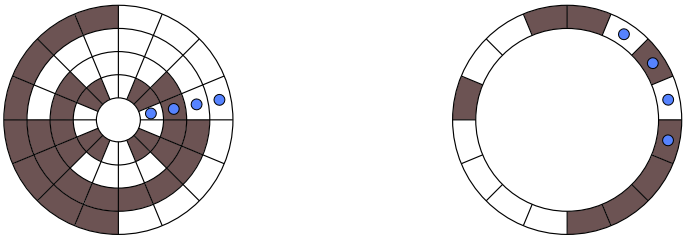
Abstract. A method for obtaining error-tolerant cyclic sequences similar to de Bruijn sequences is presented. These sequences have a number of potential applications, including use as absolute rotary encoders. This investigation is motivated by the desire to use a vision-based system to obtain the angular position of the wheels of mobile robots as they rotate about their axes. One benefit of this approach is that the actual wheel orientation is observed (as opposed to non-collocated measurements of wheel angles via encoders on the motor shaft). As a result, ambiguities from backlash are eliminated. Another benefit of this system is the ability to apply it quickly to existing systems. Several methods are developed for increasing the robustness of these encoders. An imaging simulator is used to compare the accuracy of a variety of encoding schemes subjected to several levels of image noise.

1 Introduction

The process of transmitting and obtaining state information has been studied from a variety of perspectives. A practical example of this process is determining the orientation of a wheel relative to a fixed frame. A number of rotary encoding strategies have been developed to detect both relative motion and absolute position. For absolute encoding, most rotary encoders fall into one of two categories: single-track and multitrack. Most multitrack systems rely on bits that change in parallel as the wheel turns. Single-track strategies utilize a single code, and segments of this code are often read in a serial fashion similar to a shift register. Fig. 1 provides two examples of common absolute encoders: a Gray code and a de Bruijn sequence. A general overview of rotary encoders can be found in [10], [2], and [11].

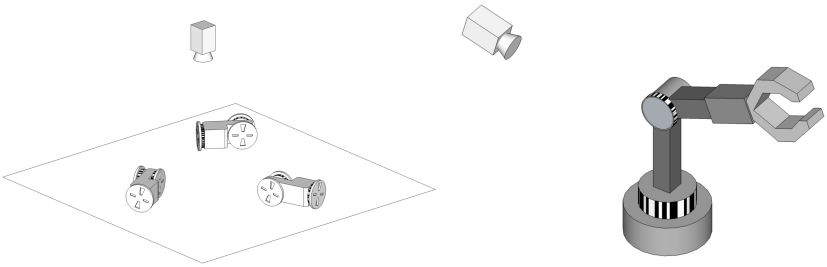
Kevin C. Wolfe · Gregory S. Chirikjian
Johns Hopkins University, Baltimore, MD, USA
e-mail: {kevin.wolfe,gregc}@jhu.edu

M.A. Hsieh and G. Chirikjian (eds.), *Distributed Autonomous Robotic Systems,*
Springer Tracts in Advanced Robotics 104,
DOI: 10.1007/978-3-642-55146-8_17,



(a) 4-bit binary Gray code (b) 4-bit binary de Bruijn sequence

Fig. 1 Above are two examples of codes that can are commonly used for rotary encoders. Small circles indicate sensor locations. The multitrack Gray code (a) is read in a parallel fashion, while the single track de Bruijn sequence (b) is read serially.



(a) (b)

Fig. 2 Examples of potential applications for vision-based encoders

In this paper, a new method for absolute rotary encoding that can be used in combination with, or in place of, shaft encoders is presented based on single-track encoding. It was developed for use with a new kind of modular reconfigurable robotic platform, the M^3 (Modular-Mobile-Multirobot) system, that is further described in [15], [8], and [14]. In this system, the orientations of the wheels are important for module-to-module docking that occurs on the wheel surfaces. Fig. 2(a) illustrates an initial deployment scenario in which an overhead camera is used to measure the wheel angles, position, and orientation of multiple robotic modules. This ability to easily capture additional state information can be important for nonholonomic motion planning and characterization of backlash. Some issues related to backlash and other uncertainty in nonholonomic robotic systems are discussed in [3], [17], and [1].

The basic approach is to print a cyclic code around the circumference of the wheel. A camera captures a visible portion of this pattern, and a decoding scheme assesses the orientation of the wheel. There are numerous advantages of this approach over traditional shaft encoding: (1) because the measurement is direct, measurement errors that accumulate between the motor shaft and the wheel due to backlash in the drivetrain are eliminated; (2) sensor/payload requirements on the robot are reduced

since sensing is external; (3) a single sensor (an overhead camera) can be used to simultaneously assess both pose and wheel angles rather than using multiple sensors; (4) this method can be used to supplement shaft encoders to quantify backlash at any given time; (5) if off-board computing is being used, data bandwidth is reduced as wheel angle information does not need to be sent from each robot to the controlling computer. Additionally, one of the most beneficial advantages of this encoding scheme is its ease of application to existing systems without the need to carefully align and mount collocated encoders and detectors.

Fig. 2(b) illustrates another potential application for the encoders presented. In this type of application, appropriate methods must be developed for dealing with occlusions. The robust nature of the encoding scheme allows for accurate readings despite minor partial occlusion. Occlusions may also be handled using multiple cameras.

The primary focus of this work is on error-tolerant cyclic sequences for use as single track rotary encoders sampled through a noisy channel, computer vision in our case. Error-tolerance here refers to error-detection and/or error-correction. Several groups have investigated error-tolerant single-track sequences that address transition error, errors that arise on the boundary between characters due to slight misalignment of detectors [12, 13, 16]. However, we are concerned with error-tolerance from a more information theoretic standpoint; we would like to improve the certainty with which we make measurements by increasing the Hamming distance between distinct positions along the code. In [6], Heiss developed error-detecting codes for multitrack encoders where the Hamming distance between adjacent locations was one and it was greater than one for all other pairs of distinct locations.

While this work was primarily developed for rotary encoding, portions of it are applicable in a variety of settings. If we consider the problem of phase synchronization of two or more systems through a noisy channel, error tolerant codes such as those described here could be utilized. Hagita et al. further describe this in [5] while also exploring and presenting methods for obtaining error-tolerant sequences similar to the ones developed here. However, they only provide a method for generating error-correcting binary sequences of period $2^{2^m-m-2}-1$ with 2^m-2-length subsequences for integer values of m [5]; such sequences may not be suitable for rotary encoders given specific design considerations.

2 Error-Tolerant Cyclic Sequences

Let $S=(a_1,a_2,a_3,\dots a_{K-1},a_K,a_1,a_2,\dots)$ be a cyclic sequence with a period of K whose elements are taken from an alphabet, $\mathbb{A}$, with q distinct elements. Now let $s_n(i)$ be a subsequence of S of length n starting at the ith element of S. For example, $s_5(3)=(a_3,a_4,a_5,a_6,a_7)$.

S is a *de Bruijn sequence* for subsequences of length n if it has a period of q^n and all possible n-length combinations of the elements of $\mathbb{A}$ appear. De Bruijn sequences have been studied and used as absolute rotary encoders. This is done by "writing" q^n consecutive elements of the sequence around a circular object, such as a wheel.

Taking n-sequential readings then allows the angular position to be determined to within $\pm\frac{\pi}{q^n}$ radians. This is illustrated in Fig. 1(b) for $q=2$ and $n=4$.

Any cyclic sequence, S, can be used as an absolute rotary encoder in a similar fashion provided that all n-length subsequences are unique. This can be formalized further if we let $d_H(s_n(i),s_n(j))$ be the Hamming distance[1] between subsequences $s_n(i)$ and $s_n(j)$. Now consider using the sequence S as an absolute encoder; $d_H(s_n(i),s_n(j))$ represents the distance between the message sent for position i and position j. For convenience we can further define the minimum Hamming distance between two distinct positions as

$$\delta(S,n) = \min\{d_H(s_n(i),s_n(j)) \,|\, i,j \le K, i \neq j\}.$$

Thus, for S to be a suitable cyclic sequence for use as an absolute encoder in the manner described above, $\delta(S,n)$ must be at least 1 (i.e., no n-length subsequence appears more than once).

If this process is viewed as transmitting an actual angular position through a noisy channel, it may be desirable to have this minimum distance be greater than 1, providing some degree of error-tolerance. If we assume that we only transmit and receive elements from $\mathbb{A}$, we are able to detect up to $\delta(S,n)-1$ errors. For error-correction, if $\delta(S,n) \ge 2e-1$, we are able to correct up to e errors. Using vision as a transmission channel, the data received may differ from $\mathbb{A}$ and thus it is less obvious the "number" of errors we are able to detect or correct; we are more concerned with the maximizing the mutual information. In either case, it should be clear that our goal is to increase $\delta(S,n)$ while decreasing n for a fixed or bounded K.

For the remainder of the paper we will refer to a cyclic sequence S as an (n,d)-sequence if $\delta(S,n) \ge d$. Using this notation it is clear that all de Bruijn sequences are $(n,1)$-sequences; as such, they are not well suited for use as encoders in the presence of noise. In Sections 3 and 4 we will explore solutions to this problem.

3 Obtaining (n,d)-Sequences and the De Bruijn Graph

A *de Bruijn graph*, $G=(V,E)$, for n-length subsequences on an alphabet, $\mathbb{A}$, of q characters is a directed graph where each vertex, $v_i \in V$, represents one of the q^n possible n-length code words. The edges connect vertices that represent possible sequential subsequences. Thus, a directed edge $(v_i,v_j) \in E$ exists if v_j can be attained by appending an element of $\mathbb{A}$ to the right of a left shifted version of v_i. Inasmuch, if

$$v_i = (a_{v_i 1}, a_{v_i 2}, \ldots, a_{v_i n}),$$

then

$$(v_i,v_j) \in E \Leftrightarrow v_j = (a_{v_i 2}, \ldots, a_{v_i n-1}, b) \quad \text{for some} \quad b \in \mathbb{A}.$$

For further information regarding de Bruijn graphs see [4].

[1] The Hamming distance for two sequences of equal length is taken to be the number of corresponding positions at which the sequences differ.

Table 1 Maximum length of binary (n,d)-sequences $(q=2)$

n	$d=1$	$d=2$	$d=3$	$d=4$
3	8	3		
4	16	4		
5	32	10		
6	64	15	7	
7	128	31	14	7
8	256	63	16	8
9	512	$\geq$100	31	11
10	1024	$\geq$188	62	22

Cycles[2] of length K such that $K \geq n$ within a de Bruijn graph are analogous to $(n,1)$-sequences. Given a cycle, C, of length $K \geq n$ from a de Bruijn graph we can construct a cyclic sequence, S_C, by sequentially taking the last element of the code words represented by its first K vertices. Therefore, let

$$C = (v_i, v_{i+1}, \ldots, v_{i+m}, v_i)$$

then

$$S_C = (a_{v_i n}, a_{v_{i+1} n}, \ldots, a_{v_{i+m} n})$$

and $C \approx S_C$. Thus, a Hamiltonian cycle[3] in G is analogous to a de Bruijn sequence.

While any cycle in G that is at least n in length represents an $(n,1)$-sequence, some cycles in G may correspond to sequences whose minimum distance is greater than 1. Given some n and $d > 1$, we could obtain (n,d)-sequences by enumerating all cycles in G using methods given in [7] and [9], and removing those that do not satisfy the constraint on d. However, for $d > 1$ the number of cycles with minimum distance d is a relatively small subset of all of the cycles in G. Consequently, enumerating all cycles may be inefficient; for example, is known that on q characters there exist $(q!)^{q^{n-1}} q^{-n}$ unique de Bruijn sequences each of which does not need to be considered when looking for such (n,d)-sequences. Therefore, a modified cycle finding algorithm was developed. This is given in Algorithm 1 which takes in a de Bruijn graph and a minimum distance and, through a breadth-first search, finds all cycles such that all vertices have a pairwise Hamming distance of at least d. Enforcement of this constraint during each search step eliminates the need to enumerate unsuitable paths or cycles and to remove unsuitable cycles at termination. Table 1 provides a summary of the maximum length (n,d)-sequences that have been found for a variety of n and d combinations using a binary alphabet.

[2] Throughout the paper, the term *cycle* will refer only to *simple cycles*, those with no repeated vertices other than the first and last.

[3] A Hamiltonian cycle is a cycle in which every vertex in a graph is visited exactly once.

Algorithm 1 Find all (n,d)-sequences

Require: De Bruijn digraph $G=(V,E)$ with vertices for subsequences of length n

procedure ALLCYCLESWITHMINDISTANCE(G,d)
  $k \leftarrow 0$
  $U \leftarrow \{\}$
  for all $v \in V$ **do**
    $k \leftarrow k+1$
    $P_k \leftarrow (v)$
    $C_k \leftarrow$ false
    $T_k \leftarrow$ true
  Note: P_k represents a path. C_k = true if P_k is a cycle. T_k = true if P_k should be further explored.
    while T(j) = true for some j **do**
      $i \leftarrow \arg\min_{1\leq j\leq k}(T_j = \text{true})$
      $y \leftarrow$ last element in P_i
      $T_i \leftarrow$ false
      for all $x \in$ CHILDREN(y,G) **do**
        if HAMMINGDIST(x,P_i) $\geq d$ **then**
          $k \leftarrow k+1$
          $P_k \leftarrow$ APPEND(P_i,x)
          $C_k \leftarrow$ false
          $T_k \leftarrow$ true
          if $x \in P_i \bigcup U$ **then**
            $T_k \leftarrow$ false
            if $x = v$ **then**
              $C_k \leftarrow$ true
            end if
          end if
        end if
      end for
    end while
    $U \leftarrow U \bigcup v$
  end for
  Return all P_j such that $C_j =$ true
end procedure

function CHILDREN(v,G)
  Let $G=(V,E)$
  Return all $w \in V$ such that $(v,w) \in E$
end function

function HAMMINGDIST(v,P)
  Return minimum Hamming distance between v and all vertices in P
end function

function APPEND(P,v)
  Let $P=(p_1,p_2,\ldots p_m)$
  $P' \leftarrow (p_1,p_2,\ldots p_m,v)$
  Return P'
end function

4 Multiple Parallel Sequences

Given a set of constraints on the length of the sequence, K, and the length of code words, n, using a single sequence may not provide enough uniqueness between all distinct positions. We can provide increased certainty by introducing additional sequences that are written in parallel to the first.

Let us assume that we want to use m sequences. For pairs of positions in one sequence whose Hamming distances are close, we can attempt to ensure that those pairwise distances are greater in the remaining sequences. Thus for two sequences $S^{(1)}$ and $S^{(2)}$, for all i and j such that $d\left(s_n^{(1)}(i), s_n^{(1)}(j)\right) = \delta\left(S^{(1)}, n\right)$ we desire $d\left(s_i^{(2)}, s_j^{(2)}\right) > \delta(S^{(2)}, n)$. To this end, we can define a $K \times K$ distance matrix $\mathbf{D}(S^{(l)}, n)$ as

$$\left[\mathbf{D}\left(S^{(l)}, n\right)\right]_{ij} = d\left(s_n^{(l)}(i), s_n^{(l)}(j)\right).$$

Using this notation we can easily sum distance matrices to obtain a distance matrix whose entries represent the total Hamming distance between two positions across all sequences.

To that end, we would like to determine the cyclic shifting or phasing that maximizes the following:

$$\max_{\mathbf{P}_1,\ldots\mathbf{P}_k \in \mathscr{P}} \left(\min_{i,j|i\neq j} \left[\sum_{l=1}^{k} \mathbf{P}_l \left(\mathbf{D}\left(S^{(l)}, n\right)\right) \mathbf{P}_l^{-1} \right]_{ij} \right) \tag{1}$$

Here the cyclic shifts are represented by shift permuation matrices $\mathbf{P}_1, \ldots \mathbf{P}_k \in \mathscr{P}$. $\mathscr{P}$ can be defined as

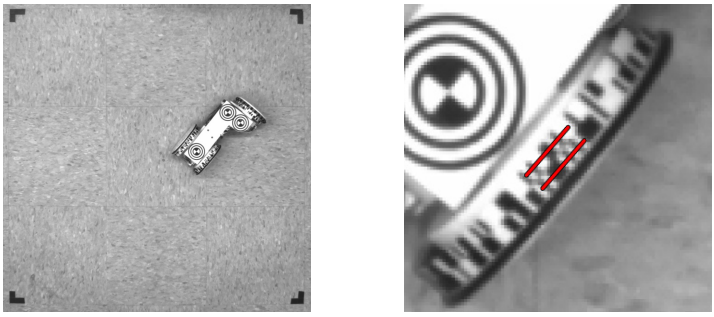
$$\mathscr{P} = \{\mathbf{P} | \mathbf{P} = (\mathbf{e}_{n-j}, \mathbf{e}_{n-j+1}, \ldots \mathbf{e}_n, \mathbf{e}_1, \ldots \mathbf{e}_{n-j-1}), \\ 0 \leq j \leq n-1\}$$

where $\mathbf{e}_i$ is the ith standard basis vector. It should be noted that the maximization in (1) can be simplified by letting P_1 equal the $K \times K$ identity matrix.

It is likely that a number of shifts will result in the same value of (1). To choose from among these we consider the standard deviation between all pairs of positions; by minimizing this standard deviation, we seek to provide a uniformity to the Hamming distance between positions.

5 Reading Wheel Position

For our application, a single overhead camera and binary sequences (i.e. $q = 2$ and $\mathbb{A} = \{0(\text{black}), 1(\text{white})\}$) are utilized. Using this camera and fiducials on the robots, we are able to locate the position and orientation of each robot. From this pose information and the known geometry of a robotic module, we determine where to read the sequences. The intensity images of the sequences are sampled along line



(a) View from overhead camera (b) Close-up of reading locations

Fig. 3 Knowing the relative position of the robot with respect to camera allows the position to be read. In (a) the full camera image is shown. The two short line segments in (b) indicate where sampling of the sequences is performed in the image.

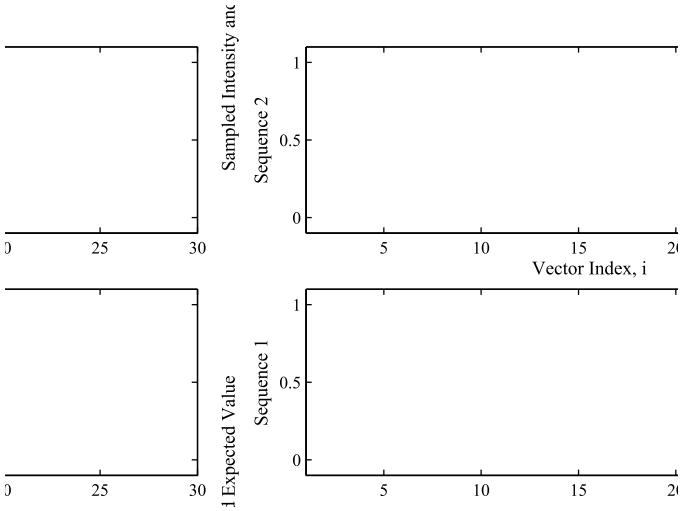


Fig. 4 Intensity readings and expected values sampled from Fig. 3.

segments; an example of this sampling is shown in Fig. 3 and Fig. 4. We take pn interpolated samples where p is the number of samples per bit. This sampling strategy enables us to increase accuracy beyond $\pm\frac{\pi}{K}$ radians to approximately $\pm\frac{\pi}{pK}$ for K-length sequences. Let $\mathbf{x}(S,t) \in [0,1]^{pn}$ be a reading taken from the image of a sequence S at time t. Let $\mathbf{y}(s_n(i)) \in \{0,1\}^{pn}$ such that

$$\mathbf{y}(s^n(i)) = (y_1, y_2, \ldots y_p, y_{p+1}, \ldots y_{pn})^T = (a_i, a_i, \ldots a_i, a_{i+1}, \ldots a_n)^T .$$

Let $\mathbf{h} = \{-0.5\}^{pn}$ be a constant vector used to shift $\mathbf{x}(S^{(l)},t)$ and $\mathbf{y}(s_n^{(l)}(i))$. This shift is needed to ensure that all vectors have the same 2-norm.

We can find the position using the summed cross-correlation across all m sequences. This can be done by letting b equal

$$b = \arg\max_i \left(\sum_{l=1}^{k} \left(\mathbf{x}(S^{(l)}, t) + \mathbf{h} \right)^{\mathsf{T}} \left(\mathbf{y}\left(s_n^{(l)}(i) \right) + \mathbf{h} \right) \right).$$

Then the angular reading can be taken as

$$\theta = \frac{2b\pi}{pK}.$$

6 Design Considerations and Imaging Simulator

For a given application, there are a number of design considerations that need to be taken into account. First, the radius and width of the cylindrical surface on which the code is to be written on will impose a number of design constraints including the number of sequences to write in parallel and the length of the sequences. These geometric factors must be coupled with the imaging resolution, position of the camera, and image quality to determine a suitable set of coding sequences. For a fixed cylinder geometry, increasing the number of sequences written in parallel will increase the Hamming distance between distinct angular positions, but will also increase the noise and blurring experienced in reading the position. Similarly, increasing sequence length reduces the size of an individual bit leading to less certainty when attempting to read the bit.

In an attempt to quantify the relative quality of various coding strategies, an imaging simulator was developed to easily test a variety of encoders. The simulator creates a virtual cylinder with binary sequences written on the outer surface. It then sets the cylinder's pose and angular position. A virtual image is captured of the cylinder at a specified resolution and common image noise is added at specified levels. The forms of image noise considered are Gaussian blurring, additive white Gaussian noise, and a reduction in the dynamic range. In addition to image noise, minor noise was added to the position of the cylinder. This has the effect of measurement uncertainty of the position of a robotic module. Once we have an image, we use the same decoding technique described in Section 5.

For a given coding scheme, the simulator was used to test reading a wheel with a given geometry in a variety of poses and at numerous noise levels. The amount of noise considered for each test was determined by a noise coefficient used to scale all of the noise sources simultaneously (see Fig. 6). The percentage of correct readings were then used to determine the effectiveness of a particular encoding. A reading was considered correct if it was within a set tolerance limit; for the test results presented in Fig. 7, the tolerance limit was set at $\pm\frac{2\pi}{K}$ for K-length sequences. Parameters were chosen that closely match those found in the $M^3Express$ system [15]. The cylinder had a radius of 55mm and a width of 12mm. The image resolution was chosen so that the each pixel represents approximately one square millimeter.

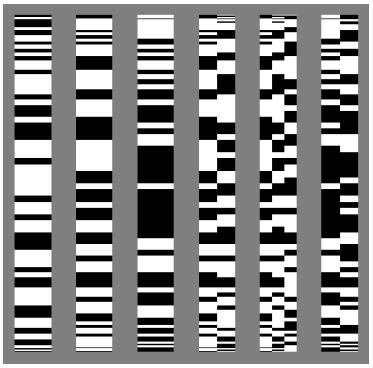


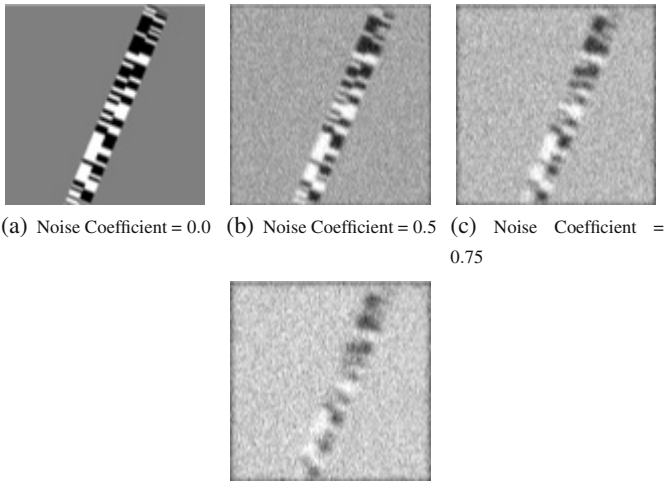
Fig. 5 Six example encoders that were tested using the image simulator. Each encoded cylinder is shown from above at high resolution. From left to right: a single $(10,1)$-sequence, a single $(8,1)$-sequence, a single $(10,2)$-sequence, two parallel $(8,1)$-sequences, three parallel $(8,1)$-sequences, and two parallel $(10,2)$-sequences.

Several examples of encoding scenarios that were tested are provided in Fig. 5. These include a single $(8,1)$-sequence, a single $(10,1)$-sequence, a single $(10,2)$-sequence, two parallel $(8,1)$-sequences, three parallel $(8,1)$-sequences, and two parallel $(10,2)$-sequences. All sequences tested have a length of 170. The $(n,1)$-sequences were generated randomly and then checked to ensure a minimum Hamming distance of 1. These $(n,1)$-sequences are similar to a standard de Bruijn sequence encoding scheme. The $(10,2)$-sequences were generated using Algorithm 1. The phasing of the multiple sequence encodings was performed as described in Section 4.

7 Discussion and Future Work

We have presented a new method for encoding and reading the angle of a wheel using vision based detection. The encoding strategy was inspired by the use of de Bruijn sequences as single track rotary encoders. In Sections 3 and 4, we describe methods for developing encoders that are more tolerant to uncertainty and noise. These methods include increasing the Hamming distance between distinct positions for single cyclic sequences and writing multiple sequences in parallel.

In addition to developing encoders, six example encoding strategies were tested using an imaging simulator described in Section 6. The results shown in Fig. 7 reveal several things. It is clear that the multi-sequence encoders perform better than single sequence encodings. This can be attributed to the redundancy provided by adding additional sequences. However, the performance difference between using one and two $(8,1)$-sequences is much more dramatic than the difference between using two and three. This may be due to blurring of the borders of bits or positioning error. As additional sequences are added, the width of each strip is reduced. For a fixed resolution image, this reduced size can lead to additional ambiguity for each bit.



(a) Noise Coefficient = 0.0 (b) Noise Coefficient = 0.5 (c) Noise Coefficient = 0.75

(d) Noise Coefficient = 1.0

Fig. 6 Four sample images demonstrating the amount of image noise introduced in the image simulator.

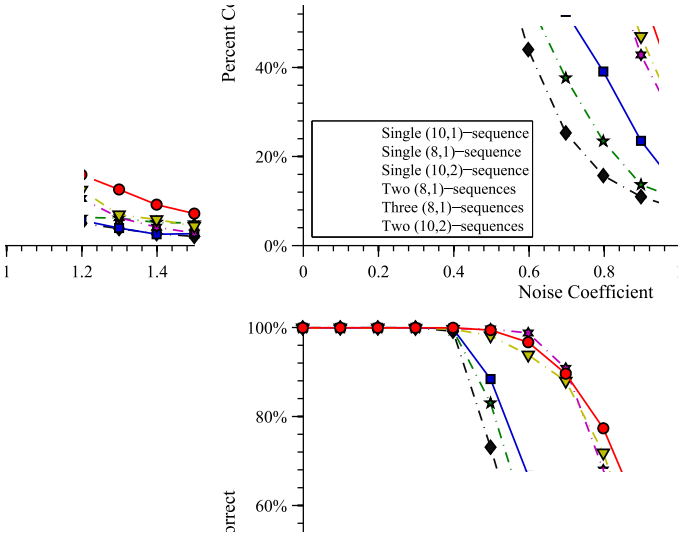


Fig. 7 The percentage of correctly read encoder positions at various noise levels for the encoder examples shown in Fig. 5.

Thus, there is a limit to the added benefit of adding additional sequences. Also evident is that for the single sequence encoders, the $(10,2)$-sequence outperforms both the $(10,1)$ and $(8,1)$-sequences. We believe that this is due to the increased Hamming distance between any two positions. Finally, it should be noted that the

encoder which employed both strategies had the highest success rate at nearly all noise levels.

While the methods described here may provide more robust encoding strategies, there is still additional cataloging of sequences for $n \geq 10$ and $d > 1$ that can be performed. Moreover, future investigation can be performed to attempt to enumerate sequences for larger alphabets, $q > 2$. These could be useful as a replacement for writing two or more codes in parallel. For example, if $q = 4$, one could imagine representing the four symbols using two bits side-by-side (i.e. the alphabet, $\mathbb{A}$, could be $\{(0,0),(0,1),(1,0),(1,1)\}$). Writing such a sequence on a wheel would then look very similar to writing two sequences in parallel; however, a greater minimum Hamming distance may be achievable for a smaller number of total readings. Finally, in addition to investigating additional encodings, future work may include testing and experimental validation of the accuracy of the encoders on the $M^3Express$ and development of strategies for handling partial occlusions.

Acknowledgements. The authors would like to graciously thank Dr. Matthew Moses for his helpful insights and discussions.

This work was suported in part by NSF grant IIS-0915542 RI: Small: Robotic Inspection, Diagnosis, and Repair.

References

1. De Luca, A., Oriolo, G., Samson, C.: Feedback control of a nonholonomic car-like robot. In: Laumond, J. (ed.) Robot Motion Planning and Control. LNCIS, vol. 229, pp. 171–253. Springer, Heidelberg (1998)
2. Everett, H.R.: Sensors for Mobile Robots: Theory and Application. A K Peters (1995)
3. Fierro, R., Lewis, F.L.: Robust practical point stabilization of a nonholonomic mobile robot using neural networks. Journal of Intelligent and Robotic Systems 20, 295–317 (1997)
4. Golomb, S.W.: Shift Register Sequences. Holden-Day (1967)
5. Hagita, M., Matsumoto, M., Natsu, F., Ohtsuka, Y.: Error correcting sequence and projective de bruijn graph. Graphs and Combinatroics 24(3), 185–194 (2008)
6. Heiss, M.: Error-detecting unit-distance code. IEEE Trans. Instrum. Meas. 39(5), 730–734 (1990), doi:10.1109/19.58616
7. Johnson, D.B.: Finding all the elementary circuits of a directed graph. SIAM J. on Computing 4(1), 77–84 (1975)
8. Kutzer, M.D.M., Moses, M.S., Brown, C.Y., Scheidt, D.H., Chirikjian, G.S., Armand, M.: Design of a new independently-mobile reconfigurable modular robot. In: Proc. IEEE International Conf. on Robotics and Automation (2010)
9. Mateti, P., Deo, N.: On algorithms for enumerating all circuits of a graph. SIAM J. on Computing 5(1), 90–99 (1976)
10. Mayer, J.R.R.: The Measurement, Instrumentation, and Sensors Handbook, ch. 6.8. CRC Press (1999)
11. Stein, D., Scheinerman, E.R., Chirikjian, G.S.: Mathematical models of binary spherical-motion encoders. IEEE/ASME Trans. Mechatronics 8(2), 234–244 (2003)
12. Tomilnson, G.H.: Absolute type shaft encoder using shift register sequences. Electronics Letters 23(8), 398–400 (1987)

13. Tomlinson, G.H., Ball, E.: Elimination of errors in absolute position encoders using m-sequences. Electronics Letters 23(25), 1372–1374 (1987)
14. Wolfe, K.C., Kutzer, M.D., Armand, M., Chirikjian, G.S.: Trajectory generation and steering optimization for self-assembly of a modular robotic system. In: Proc. IEEE International Conf. on Robotics and Automation (2010)
15. Wolfe, K.C., Moses, M.S., Kutzer, M.D., Chirikjian, G.S.: M3express: A low-cost independently-mobile reconfigurable modular robot. In: Proc. IEEE International Conf. on Robotics and Automation (2012)
16. Yamaguchi, K., Imai, H.: Periodic sequences for absolute type shaft encoders. In: Sakata, S. (ed.) AAECC 1990. LNCS, vol. 508, pp. 36–45. Springer, Heidelberg (1991)
17. Zhang, Q., Shippen, J., Jones, B.: Robust backstepping and neural network control of a low-quality nonholonomic mobile robot. International Journal of Machine Tools and Manufacture 39, 1117–1134 (1999)