Tesis de Licenciatura

Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

"Cómo embeber Glyphs en Códigos de Barras"

Alumno: Hernán Martín Podolsky
L.U.: 659/88

Alumno: Eduardo Miguel Raichman
L.U.: 231/89

Director: Eduardo J. Rodríguez

Año 2003

Resumen

Introducidos masivamente durante los años 80, los códigos de barras se han esparcido desde supermercados a tiendas departamentales, las fábricas, lo militar, la industria de la salud, la industria del seguro, etc. [Pav90] A comienzos de la década del 90 la Asociación de Alimentos y Drogas (F.D.A.) de los Estados Unidos puso a consideración pública una propuesta de obligatoriedad de uso de códigos de barras en las medicaciones y los brazaletes de los pacientes, de manera tal que en los hospitales los enfermeros puedan escanear ambos y asegurarse el suministro de la dosis correcta de la droga indicada por el médico; el objetivo es evitar los errores que han costado miles de víctimas a lo largo de los últimos años.

A pesar que nuevas simbologías con mayor poder expresivo han aparecido en los últimos 15 años, la penetración de la tecnología de código de barras en nuestras vidas es tan profunda que un cambio en su utilización sería muy costoso para las compañías y produciría un sinfín de problemas en la identificación de los productos. Así las cosas, tendremos que convivir con esta simbología poco expresiva durante mucho años más.

Sin embargo no todo parece perdido. Este trabajo muestra que es posible introducir una tecnología de mayor poder expresivo denominada Dataglyphs, propietaria de Xerox, de manera embebida dentro de los códigos de barras tradicionales, de tal manera que no se vea alterado su escaneo por parte de los lectores de códigos de barras de mercado, y que a su vez permita la decodificación, por parte de otros dispositivos o mecanismos, de la información expresada en forma embebida dentro de los códigos de barras.

Abstract

The bar code, which had been massively introduced during the 80's, has spread from supermarkets to department stores, factories, the military, the health industry, the insurance industry, etc. [Pav90] At the beginning of the 90's the Food and Drug Administration of The United States presented a proposal for public consideration to make the use of bar codes on medication and patients I.D. bracelets obligatory. This would allow the hospital nurses to scan both the patients and the medication in order to guarantee the correct dosage of the drug indicated by the physician. The objective of this measure would be the elimination of errors which have caused thousands of victims during the last few years.

In spite of the fact that new symbologies with increased expression capacity have appeared during the last 15 years, the penetration of the bar code technology in our lives is so deep, that a change in its use would be extremely expensive for the companies and would produce endless problems in the identification of products. Being this the case, we shall have to make due with this less expressive symbology for many more years.

Nevertheless, not all is lost. This thesis demonstrates that it is possible to introduce an enhanced expression capacity technology called Dataglyphs, owned by Xerox, embedded within the traditional bar codes in such a way that its scan reading by the common everyday market bar code readers is not altered, and at the same time permits the de-codification of information expressed in embedded form within the bar code by other devices or mechanisms.

Contenido

1. Introducción

Imperceptiblemente, los códigos de barras nos rodean. Están en los productos del supermercado, en los libros, en las tapas de los discos compactos, en los productos electrónicos, en los diarios y en las revistas. Los hay en las tarjetas de fidelidad, en las boletas de impuestos, en los pasajes aéreos, en embalajes, hasta en las estanterías de los depósitos.

Los códigos de barras proveen un excelente mecanismo de identificación de muchas cosas. Permiten por ejemplo, facilitar la tarea de un cajero de supermercado, o de un cliente al averiguar el precio de un producto.

Las décadas del 80 y del 90 han visto la introducción de muchas simbologías nuevas, y un debate permanente las compara con las tradicionales y con esquemas para codificar información impresa en un sustrato. [Pav90]

En los últimos años también se han visto demandas para incrementar la densidad de la información codificada y poder llevar una gran cantidad de información del producto adosada al mismo, es decir, encontrar un mecanismo portable de almacenar una importante cantidad de información junto al producto, o lo que se ha denominado *portable data file* (archivo portable de información) [Pav90]

Se puede ver la distinción entre las dos formas en el siguiente ejemplo. El código de barras de un ítem de supermercado consiste de 11 dígitos los cuales representan un número de identificación pero no una descripción del producto. Para saber el precio, entre otras cosas, se debe acceder a una base de datos utilizando el número del código de barras. Una alternativa sería usar un código de barras más largo (u otra codificación) para almacenar toda la información relevante, como el precio, nombre del producto, nombre del fabricante, peso, datos de inventario, fecha de vencimiento, etc. Eso constituiría un *portable data file*, porque la información relevante puede ser recuperada sin acceder a la base de datos. [Pav90]

Sin embargo, todos los intentos de utilización de portable data files no fueron masivamente exitosos debido a que su implantación involucra el reemplazo de los dispositivos de lectura de códigos de barras existentes de manera de poder leer otros códigos de gran densidad de información.

Todo comerciante, compañía o productor desea preservar la infraestructura instalada especialmente si no está obsoleta o fuera del mercado. Una alternativa válida para almacenar gran cantidad de información y preservar la infraestructura de hardware y software es utilizar un *código embebido* dentro de los códigos de barras convencionales. De esta manera se permite la lectura del código subyacente a través de lectores especiales y no se altera la lectura de los códigos de barras tradicionales.

Un *código embebido* es un código dentro de otro cuya lectura, realizada de cierta forma particular, nos devuelve una información no necesariamente igual a la información que resulta de la lectura del código que lo contiene [Hec01]. Por ejemplo, en la frase "**Co**mo **di**jo **Go**ycoechea **em**piezo **be**biendo **bi**rra **do**rada", leyendo las dos primeras letras de cada palabra se puede leer otra frase *embebida* dentro de la original. En general, para leer un código embebido hay que establecer un protocolo a través del cual interpretar el código subyacente; en este tipo particular de código embebido el protocolo es: "leer las dos primeras letras de cada palabra".

La utilización de un código embebido dentro de un código de barras convencional trae aparejada cierta problemática que es necesario manejar. Dependiendo de cómo se construya el código embebido, el código de barras original puede ver afectada su calidad y consecuentemente podrán producirse lecturas o decodificaciones erróneas de su contenido.

Una alternativa es trabajar sobre la redundancia vertical del código de barras, y sobre la capacidad de hilvanar partes de código leídas correctamente a través de las repetidas lecturas de los dispositivos de lectura. [Pav90] Esto no significa que el operador deba permanecer un apreciable mayor tiempo intentando leer el símbolo, sino que la forma normal de operación del dispositivo lector es realizar varias lecturas en un lapso breve hasta lograr la decodificación completa de ese símbolo; en este caso el dispositivo de lectura se ocupará de encadenar las decodificaciones parciales que vaya logrando utilizando para ello las sucesivas lecturas que normalmente realice. [Pav90]

Otra alternativa es tratar de utilizar tamaños suficientemente pequeño de símbolos del código embebido de modo tal que sean imperceptibles para el lector de código de barras convencional, esto es, aprovechar los tamaños menores a los que puede distinguir el dispositivo lector de códigos de barras.

La tecnología de Dataglyphs, perteneciente a Xerox PARC, tiene muy buenas cualidades para trabajar como código embebido de los códigos de barras, ya que sus elementos son pequeños y visualmente no obstructivos. La estrategia entonces, será hacer el mejor uso de las ventajas que ofrece el código de barras tradicional, y embeber los Dataglyphs en él.

En el capítulo 2 se hará una breve introducción a la codificación de la información de los códigos de barras convencionales, su historia, las simbologías, los estándares, los dispositivos de lectura e impresión, la calidad de los símbolos, y la seguridad de la información. También se presentarán algunas Medidas de la Información de las distintas simbologías.

En el capítulo 3 se hará una introducción a la tecnología de DataGlyphs de Xerox, mostrando algunos datos técnicos y un pequeño detalle de las funciones de las librerías de Xerox utilizadas en este trabajo.

En el capítulo 4 se detallará la resolución del problema objeto del presente trabajo. Se describirá globalmente primero y en detalle después, el circuito de procesos que se han desarrollado para su resolución.

En el capítulo 5 se presentarán resultados comparativos entre la integración de tecnologías desarrollada en este trabajo y otras tecnologías de interés, a través de indicadores de *expresividad*, de *eficiencia*, y de *densidad*.

En el capítulo 6 se expresarán las conclusiones al presente trabajo indicando los argumentos más relevantes a favor de los resultados logrados.

Finalmente, en el capítulo 7 se presentarán ideas para trabajos futuros.

2. Breve Revisión de Códigos de Barras

2.1 Definiciones

Un *código de barras* es una disposición en paralelo de barras verticales separadas por espacios que contiene información en el ancho mismo de las barras y los espacios, pero no en su altura, es decir, codifican información a lo largo de una dimensión. Representa un método simple y preciso de codificar información que puede ser leída por algún dispositivo óptico que luego envía esa información a una computadora. [Pav90] [Pal91]

La definición anterior corresponde a los códigos de barras más difundidos que son denominados "de una dimensión". También existen otros tipos de códigos de barras, de menor utilización, conocidos como "de dos dimensiones". En el presente trabajo sólo serán de interés los códigos de barras "de una dimensión".

Un *módulo* es la mínima unidad de expresión, o medida de ancho, de una barra o espacio. Se dice que una barra o un espacio ocupa 'n' módulos, donde 'n' no necesariamente es un numero entero.

Una disposición específica de barras separadas por espacios se denomina *símbolo*. Un símbolo expresa un determinado mensaje o texto, y lo hace en el tamaño del ancho de las barras y espacios. Este texto, sin embargo, no es universal ya que el tamaño de las barras y espacios para cada caracter del texto está determinado en lo que se denomina simbología. Por lo tanto, una *simbología* es como un idioma, es un conjunto de reglas que expresan los tamaños de las barras y espacios para cada letra de esa simbología. Por supuesto, existen muchas simbologías así como existen muchos idiomas. Y un símbolo es, entonces, un mensaje expresado en una simbología determinada. También se dirá que un símbolo es la "visualización física" o la "versión impresa" de un código de barras, aunque en el lenguaje corriente se denomina "código" o "código de barras" al símbolo en sí.

Para completar la definición anterior se dirá que una simbología se define indicando también cuantas barras y espacios hay en cada elemento o letra, y en ocasiones cuantos de estos elementos son anchos, si tiene caracteres de arranque/parada, como son las quiet zones. Por ejemplo, en la simbología denominada UPC cada elemento tendrá dos barras y dos espacios distribuidos en 7 módulos. En otra simbología denominada Code 39 cada elemento tendrá cinco barras y cuatro espacios por cada elemento, mientras que tres de esos nueve elementos serán anchos. La cantidad de módulos que ocupa cada elemento depende de la relación ancho/angosto (ratio) determinada a priori en la simbología. [Pal91]

2.2 Historia

Mucho antes que los códigos de barras y los dispositivos lectores escáners fueran finalmente inventados, las tiendas o almacenes necesitaban imperiosamente un sistema similar a ellos. Las tarjetas perforadas fueron inventadas hacia finales del siglo XIX, y constituyeron una de las primeras formas de identificación de productos en dichos tipos de comercio.

En 1932, un estudiante de negocios llamado Wallace Flint, escribió una Tesis de Posgrado en la cual avizoró un supermercado donde los clientes perforarían tarjetas para marcar sus selecciones de compra. En la caja, la insertarían en un lector, el cual activaría la maquinaria que le traería las compras en cajas cerradas. Los almacenes también tendrían un registro de lo que se estaba comprando.

El problema era que el equipamiento de lectura de tarjetas no estaba ampliamente difundido y era tremendamente caro. Aún si Estados Unidos no hubiese estado en medio de "La Gran Depresión" - nombre como se conoció la depresión económica de los Estados Unidos a comienzos de la década del '30-, el esquema de Flint no hubiese sido realista, sólo preanunció lo que vendría. [Pal91]

El primer paso hacia los códigos de barras actuales sucedió en 1948, cuando Bernard Silver, un estudiante graduado escuchó una conversación en el hall del Instituto de Tecnología Drexel de Filadelfia. El presidente de una cadena de comidas estaba discutiendo con una de las autoridades del Instituto la idea de emprender una investigación sobre captura automática de la información de productos en la caja. La autoridad desestimó el requerimiento, pero Bob Silver le mencionó la conversación a su amigo Norman Joseph Woodland, un estudiante graduado de 27 años y profesor en Drexel. El problema fascinó a Woodland.

Su primera idea fue usar patrones de tinta que brillaran bajo la luz ultravioleta, y ambos construyeron un dispositivo para testear el concepto. Funcionó, pero encontraron problemas que iban desde la inestabilidad de la tinta hasta los costos de impresión. No obstante, Woodland estaba convencido que tenía una idea "trabajable". Tomó algunas ganancias que había obtenido en el mercado financiero, se fue de Drexel, y se mudó al departamento de su abuelo en el estado de Florida para buscar soluciones. Luego de varios meses de trabajo logró obtener un código de barras lineal, usando elementos de dos tecnologías establecidas: pistas de sonido de películas y el código Morse.

Woodland, ya retirado, recordó respecto al código Morse, "sólo extendí los puntos y los guiones e hice las líneas angostas y anchas". Para leer la información, él usó el sistema de sonido que usó Lee de Forest en sus películas en la década de 1920. De Forest había impreso un patrón con grados variables de transparencia sobre el borde del film, luego iluminaba una luz a través de dicho film mientras la película corría. Un tubo sensitivo del otro lado traducía los cambios de brillo en ondas eléctricas, las cuales a su vez eran traducidas a sonido por los parlantes. Woodland planeó adaptar ese sistema reflejando luz desde las líneas anchas y angostas usando un tubo similar para interpretar los resultados.

Así, Woodland llevó su idea a Drexel, donde comenzó a trabajar en una aplicación que pudiera patentarse. Decidió reemplazar sus líneas anchas y angostas por círculos concéntricos, así podría ser escaneado en cualquier dirección. Esto se hizo conocido como el "código del ojo de toro" –"bull's eye code". Mientras tanto, Silver investigó que forma debían finalmente disponer los códigos. Ambos presentaron una patente el 20 de Octubre de 1949.

En 1951 Woodland consiguió un trabajo en IBM, donde tuvo la esperanza que su esquema floreciera. Al año siguiente, él y Silver lograron construir el primer lector real de códigos de barras. El dispositivo era del tamaño de un escritorio y tenía que ser envuelto en plástico negro para evitar que entrara la luz ambiente. Estaba basado en dos elementos clave, una bombita incandescente de 500 watts como fuente de luz, y un tubo foto-multiplicador RCA 935, diseñado para sistemas de sonido de películas, cumpliendo la función de lector.

Woodland luego cambió el RCA 935 por un osciloscopio. Luego tomó una pieza de papel marcada con líneas y la movió a través de un haz delgado que emanaba de una fuente de luz. El haz reflejado era dirigido al tubo. En cierto punto el calor procedente de la poderosa bombita comenzaba a derretir el papel. A pesar de esto, Woodland obtuvo lo que quería: a medida que el papel se movía, la señal iba al osciloscopio. Él y Silver habían creado el dispositivo que podría leer electrónicamente el material impreso.

Igualmente, no estaba del todo claro como transformar esta respuesta electrónica cruda en una forma utilizable. Las computadoras primitivas de la época eran complicadas de operar, podían sólo realizar cálculos simples, y en cualquier caso eran del tamaño de un living. La idea de instalar miles de ellos en supermercados habría sido pura fantasía. Sin una forma barata y conveniente de registrar la información desde los códigos de Woodland y Silver, su idea no sería más que una curiosidad.

Luego estaba esa bombita de 500 watts. Creaba una cantidad enorme de luz, de la cual sólo una pequeña fracción era leída por el tubo RCA 935. El resto era liberado como calor desperdiciado, que además era caro y nada cómodo. Según Woodland, era tan potente que podía ocasionar daño ocular. Los inventores necesitaban una fuente de luz que pudiera focalizar una gran cantidad de luz en un espacio muy pequeño. Hoy eso se puede asociar sencillamente con luz láser, pero en 1952 los lásers no existían. En retrospectiva, los códigos de barras eran claramente una tecnología cuyo momento no había llegado, y tampoco estaba cerca.

Pero Woodland y Silver percibieron el potencial que tenía. En octubre de 1952 su patente le fue aprobada. Woodland se quedó en IBM y a fines de esa década convenció a su compañía de contratar un consultor para evaluar códigos de barras. El consultor estuvo de acuerdo en que ellos tenían grandes posibilidades pero dijo que requerirían tecnología que estaba aún a por lo menos 5 años de distancia. En ese momento casi la mitad del tiempo otorgado por la patente a Woodland y Silver había transcurrido.

IBM ofreció varias veces comprar la patente, pero los inventores consideraron que era muy poco el dinero ofrecido y no lo creyeron conveniente. En 1962 Philco logró acordar un precio, y ellos la vendieron. Al año siguiente Silver murió a la edad de 38 años. Philco luego vendió la patente a RCA.

Luego de esto, los siguientes avances en el manejo de la información saldrían de la industria del ferrocarril.

Los vagones de carga del ferrocarril se movían a través de todo Estados Unidos, y a menudo eran prestados entre las compañías. Llevar control de ellos era uno de los temas más complejos que la industria del ferrocarril tenía y a mediados de la década del 60 atrajo la atención de David Collins. Collins obtuvo una maestría en el MIT en 1959 e inmediatamente fue a trabajar a Sylvania Corporation, la cual estaba tratando de encontrar aplicaciones militares a una computadora que había construido. Recordando Collins su trabajo de antes de recibirse en la compañía Pennsylvania Railroad y sabiendo que existía la necesidad de identificar vagones automáticamente y luego manejar la información obtenida. La computadora de Sylvania podía hacer esto último. Todo lo que Collins necesitaba era una forma de identificar los vagones. Alguna forma de etiqueta codificada parecía ser la opción más barata y sencilla.

Estrictamente hablando las etiquetas que Collins había pensado no eran códigos de barras. En lugar de trabajar sobre barras negras o anillos ellos usaron grupos de rayas azules y naranjas hecha de un material reflectivo que podía estar armado para representar dígitos del 0 al 9. Cada coche fue numerado con 4 dígitos para identificar la compañía a la que pertenecía y 6 dígitos para identificar el vagón en sí mismo. Cuando los vagones iban dentro de un corral, los lectores disparaban un haz de luz de color sobre los códigos e interpretaba los reflejos. La compañía Boston & Maine llevaron a cabo el primer test del sistema en algunos de sus vagones en 1961. Hacia 1967 muchos de los trenes habían sido marcados y ya se había adoptado como estándar nacional de un sistema de codificación. Todo lo que quedaba para las compañías era comprar e instalar el equipamiento.

Collins previó aplicaciones para la codificación automática más allá de la industria del ferrocarril y en 1967 le comentó la idea a sus jefes en Sylvania. Recuerda haber dicho, "lo que deberíamos hacer ahora es desarrollar un equivalente en pequeñas líneas negras y blancas para llevar control de los convoyes y para todo aquello que se mueva". En un clásico caso de una compañía corta de vista, rechazaron ayudarlo financieramente. Ellos dijeron, "No queremos invertir más. Tenemos este gran mercado – el del ferrocarril -, vamos y hagamos plata de él". Collins renunció y fundó con otra persona Computer Identics Corporation.

Sylvania nunca vio ganancias como resultado de su intervención en la industria del ferrocarril. Los transportistas comenzaron a instalar dispositivos lectores escáners en 1970 y el sistema funcionó como se esperaba pero era muy caro. A pesar que las computadoras fueron siendo cada vez más pequeñas, rápidas y baratas aún eran muy caras para ser económicas en las cantidades requeridas. La recesión de mediados de los 70 quebró el sistema así como la agitación de las bancarrotas de las compañías de trenes quebró los presupuestos de la industria. Sylvania fue abandonado con un elefante blanco.

Mientras tanto, Computer Identics prosperó. Su sistema usaba lásers, que a finales de los 60s estaban poniéndose económicamente alcanzables. Un haz de luz láser hecho de helio y neón de una cantidad de "miliwatts" coincidió con el trabajo hecho por Woodland, quién dejó de lado la bombita de 500 watts. Una delgada luz moviéndose sobre el código de barras sería absorbida por las líneas negras y reflejadas por las blancas, dando a los sensores del escáner una clara señal on/off. Los lásers podrían leer códigos de barras en cualquier lado desde 3 pulgadas a varios pies de distancia, y podrían barrer hacia ambos lados cientos de veces por segundo, dando al lector muchas vistas de un único código desde diferentes ángulos. Esto probaría ser una gran ayuda descifrando rajaduras en etiquetas desgarradas.

En 1969, Computer Identics instaló calladamente sus dos primeros sistemas, probablemente el primer verdadero sistema de código de barras en la historia. Uno fue en una planta de General Motors en Michigan donde fue utilizado para monitorear la producción y distribución de ejes de automóviles. El otro se instaló en una compañía distribuidora propiedad de la General Trading Company en New Jersey, para ayudar a dirigir los envíos a las bahías correctas de carga. Hasta este punto los componentes habían sido construidos a mano, hechos con elementos caseros. Ambos sistemas se basaban en sistemas de códigos de barras muy simples que codificaban sólo dos dígitos de información. Igualmente, en General Motors y en General Trading no había necesidad de codificar más que eso.

El éxito Computer Identics probó el potencial de los códigos de barras en entornos industriales. Sin embargo, los almacenes o supermercados debían ser los impulsores concluyentes de esta tecnología. A comienzos de la década del 70 la industria permitió impulsar la maduración comercial de la tecnología que Woodland y Silver habían soñado y que Computer Identics probó era factible.

RCA todavía estaba moviéndose para asistir a la industria. Los ejecutivos de RCA habían participado de una reunión del mercado de los almacenes en 1966 donde había sido urgido el desarrollo de los códigos de barras, y ellos olfatearon la existencia de un nuevo negocio. Un grupo especial fue a trabajar a un laboratorio de RCA en New Jersey, y la cadena de almacenes Kroger se ofreció como conejillo de indias. Así, a mediados de 1970, un consorcio de la industria estableció un comité ad-hoc para introducirse en los códigos de barras. El comité estableció guías para el desarrollo de los códigos de barras y crearon un subcomité de selección de símbolo para ayudar a estandarizar esta aproximación.

El comité trabajaba bajo algunos lineamientos básicos. Para hacerle la vida más simple al cajero los códigos de barras deberían ser legibles desde casi todos los ángulos y a un amplio rango de distancias. Para que puedan ser reproducidas en una escala masiva las etiquetas deberían ser baratas y fáciles de imprimir. Para ser económicamente convenientes los sistemas automáticos de caja deberían repagarse en dos años y medio. A mediados de los años 70 un estudio de McKinley y Compañía predijo que la industria ahorraría 150 millones de dólares por año adoptando esos sistemas. Por otra parte, el empuje final lo darían los minoristas si los ahorros fuesen verdaderamente cuantificables. Algunos de éstos eran poder cobrarle a los clientes en la caja al doble de la velocidad que si utilizaran el equipamiento normal, sin incrementar la cantidad de empleados.

Así, mientras los primeros sistemas de códigos de barras automatizarían la caja, no serían útiles para la realización del inventario, porque al comienzo muy pocos productos vendrían etiquetados, habría que esperar hasta que fuese ampliamente difundido.

En la primera mitad del año 1971 RCA hizo una demostración del sistema de código de barras "ojo de toro" en un encuentro de la industria de los almaceneros. En esa reunión estaban algunos ejecutivos de IBM que presenciaron asombrados el evento y quedaron preocupados por estar perdiéndose un gran mercado potencial. Uno de los ejecutivos presentes recordó que el inventor del código de barras aún estaba en el staff de la compañía. Rápidamente Woodland (cuya patente había vencido en 1969) fue trasladado a North Carolina y jugó un papel preponderante en el desarrollo de la versión más popular e importante de la tecnología, el código UPC (Universal Product Code)

Mientras tanto, RCA continuó empujando su tecnología del código "ojo de toro". En julio de 1972 comenzaron un test por 18 meses en un almacén Kroger de Cincinatti. Se encontraron algunos problemas de impresión y dificultades de escaneo, que limitaban la utilidad de ese código. En la impresión, algunas veces se desplazaba tinta en la dirección de impresión y esto no permitía luego su correcto escaneo. Por otra parte, si esto le sucedía a UPC sólo afectaba una parte del código y la información no se perdía. La competencia duró un tiempo, pero al final UPC se impuso y la industria adoptó naturalmente al técnicamente elegante representante de IBM. La adopción de UPC, el 3 de abril de 1973, fue el hito más importante de la historia de la logística moderna. Así, los códigos de barras pasaron de ser una curiosidad tecnológica a un negocio monstruoso.

Antes de UPC, varios sistemas estaban funcionando alrededor del mundo, en negocios, en bibliotecas, y en fábricas, cada uno con su propio sistema propietario. La estandarización iba a llevarse varios millones de dólares en equipamiento de impresión, escaneo y etiquetado. Además, para que la estandarización resultara exitosa tuvieron que inventarse nuevos tipos de tinta y sustratos de manera de reproducir el código con las tolerancias exactas que requiere.

Dos desarrollos tecnológicos de los 60s hicieron finalmente que los escáners fueran económicamente viables. El primero fue el desarrollo de los lásers baratos. El segundo fue la invención de los circuitos integrados. El 26 de julio de 1974, en un supermercado Marsh de Ohio se vendió el primer paquete de chicle con código de barras con la ayuda de un escáner. El crecimiento en el uso de escáners fue inicialmente lento. Un mínimo del 85% de todos los productos debía venir con código de barras para que el sistema se pague. A finales de los años 70 las ventas de los sistemas comenzaron a despegar. En 1978, menos del 1% de las tiendas minoristas en todo Estados Unidos tenían escáners. En 1981 era alrededor del 10%, en 1984 era del 33%, y hasta el día de hoy el 60% están equipadas.

La tecnología ha progresado mucho, y ha ido hacia otras industrias y organizaciones. Los investigadores, por ejemplo, han montado pequeños códigos de barras en abejas para poder estudiar sus hábitos de comportamiento. Las Fuerzas Armadas de los Estados Unidos utilizan grandes etiquetas de códigos de barras para etiquetar barcos en una estación de almacenaje. Los pacientes en los hospitales usan códigos de barras con sus IDs en brazaletes. También se ven en partes de camiones, documentos comerciales, cajas de envíos, y en corredores de maratón. Federal Express es probablemente el usuario individual de códigos de barras más grande del mundo; sus etiquetas utilizan el código conocido como Codabar. Más adelante han ocurrido modernos refinamientos del código original, incluyendo el código EAN, desarrollado por Joe Woodland, que incluye un par adicional de dígitos, y se está convirtiendo en el código más utilizado a nivel mundial. Existen otros códigos como Code 39, Code 16K, o Interleaved 2 of 5, algunos contienen sólo números, otros contienen números y letras.

Woodland nunca se volvió rico con los códigos de barras, a pesar que fue galardonado con la Medalla Nacional de la Tecnología en Estados Unidos en 1992 por el presidente de ese país. Su invención creó una nueva forma de hacer negocios en todo el mundo, y dice mucho acerca de pequeñas personas con una gran visión. [WWW2]

2.3 Simbologías

Como ya se ha expresado, existen diversas simbologías. Muchas de ellas se han afianzado como forma unívoca de identificación de productos en alguna industria o mercado, como por ejemplo el código denominado UPC lo ha hecho en comercio minorista de EEUU desde 1973.

Los aspectos que caracterizan a las distintas simbologías son los siguientes:

- *Juego de Caracteres*. Cada simbología define un conjunto de caracteres de datos que puede codificar. Algunas simbologías sólo pueden codificar información numérica, otras pueden codificar información alfanumérica, mientras que otras soportan 128 caracteres ASCII
- *Tipo de Simbología*. Las simbologías pueden ser discretas o continuas. En una simbología *discreta*, cada caracter puede decodificarse independientemente de sus caracteres adyacentes ya que cada caracter está separado de sus vecinos por espacios en blanco de tamaño específico llamados *Inter-Character Gaps* y que no expresa información alguna. En una simbología *contínua* un caracter termina donde empieza su vecino inmediato. La no existencia de espacios de separación achica el tamaño físico necesario para expresar cierta cantidad de información. Ésta mejora en la densidad puede eventualmente condicionar la tecnología utilizada para imprimir el símbolo. La seguridad relativa de los datos de las simbologías no está afectada por el hecho que sean discretas o continuas
- *Número de elementos anchos*. Existen dos tipos de simbologías, las que emplean sólo dos anchos de elementos (ancho y angosto) y las que emplean múltiples anchos, llamadas también 'delta'. [Pav90] La simbología Code 39, por ejemplo, es de las que emplean sólo dos anchos; la simbología UPC utiliza múltiples anchos, lo cual no es una regla base de la simbología sino que se desprende de las posibles combinaciones de dos barras y dos espacios en 7 módulos, y así se forman los múltiples anchos. La mayoría de las simbologías de múltiples anchos son modulares, es decir que la longitud de cada caracter está subdividida en un número predeterminado de módulos, y el ancho de las barras o espacio es siempre un número entero de módulos. La mayoría de las simbologías de múltiples anchos son continuas y a menudo son decodificadas utilizando algoritmos que miden la distancia entre ejes similares de barras adyacentes en lugar de medir realmente el ancho de cada barra o espacio. Se muestra un ejemplo en la Fig 2-1.

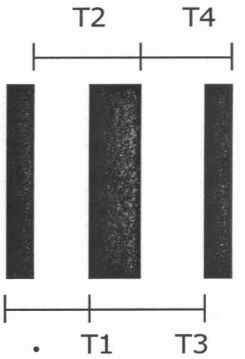


Fig 2-1. Distancias T.

Aquí T1 y T3 son las distancias entre los ejes iniciales de las barras

- *Longitud fija o variable*. Algunas simbologías, por definición estructural, sólo codifican mensajes de una longitud fija. Otras deben ser utilizadas con una longitud fija debido a consideraciones de seguridad de la información, mientras que otras pueden ser utilizadas para codificar información de longitud realmente variable
- *X* y *Z*. *X* es el término utilizado para describir el ancho nominal (o pretendido) del elemento más angosto de un código de barras, ya sea una barra o un espacio. *X* normalmente corresponde a 1 módulo de longitud. Cuando se examina un símbolo, es común medir y calcular el ancho promedio de los elementos más angostos del símbolo, que debido a defectos de impresión y otros factores no es exactamente X, entonces es llamado *Z*. Por convención, X y Z están expresados en 'mils'. (milésimas de pulgada)
- *Densidad*. Las simbologías de códigos de barras difieren en la cantidad de información que puede ser codificada en cierta longitud. Para permitir comparaciones significativas, el valor X es utilizado para estudiar densidades relativas. La densidad, generalmente está especificada para los caracteres de datos, pero para medir la densidad de un símbolo hay que considerar también los caracteres de control (arranque/parada, quiet zone, y caracteres de chequeo). El concepto de densidad puede ser extendido a estudiar la eficiencia en área o superficie de una simbología
- *Auto chequeo*. Una simbología tiene auto chequeo si un solo defecto de impresión no causa la mala interpretación de un caracter por otro de la misma simbología
- *Caracter de arranque, caracter de parada*. Los caracteres de arranque y de parada son dos patrones particulares de barras y espacios que son puestos al inicio y al final del símbolo para que el dispositivo lector sepa donde comienza y donde termina el símbolo; algunas veces también indican la dirección de lectura
- *Quiet zones*. Son las zonas vacías (en blanco) que se encuentran antes del caracter de arranque y después del caracter de parada. Normalmente deben ser de un cierto ancho, y eventualmente permiten separar símbolos adyacentes
- *Caracter de verificación*. Un caracter de verificación es un caracter puesto en una posición predeterminada del símbolo y cuyo valor está basado en una relación matemática entre los otros caracteres del símbolo. Es utilizado por el dispositivo lector para validar que ha sido decodificada la información correcta
- *Bidireccional*. Una simbología es bidireccional si un símbolo puede ser leído por el dispositivo de izquierda a derecha o viceversa sin afectar la información decodificada. Casi todas las simbologías en uso actualmente son bidireccionales
- *Auto seguimiento (Self-Clocking)*. Los dispositivos lectores necesitan información de referencia para poder medir las posiciones relativas de los ejes de los elementos. Algunas simbologías antiguas incluían una suerte de marcas de seguimiento. Las simbologías modernas son diseñadas de forma tal que los escáners no requieran esa ayuda para calcular los anchos [Pal91]

Una simbología muy conocida es UPC. El estándar UPC, denominado UPC-A, codifica un número de 12 dígitos. El primero es llamado "número del sistema". La mayoría de los productos tiene un "1" o un "7" en esta posición. Los dígitos del segundo al sexto representan el número del fabricante, y es asignado por un organismo rector lo cual evita códigos duplicados. Los caracteres del séptimo al onceavo corresponden a un número que identifica cada producto de ese fabricante, lo que se ha dado en llamar "número del producto". El doceavo caracter es el "dígito verificador", resultante de un algoritmo que involucra a los 11 números previos. La industria editorial ha agregado números suplementarios (de dos a cinco) al final del símbolo UPC-A, utilizados por lo general para la fecha de publicación o el precio. También existen las versiones UPC-E que es utilizada para etiquetas chicas y con muchos números 0 entre medio, y la UPC-D que es muy raramente utilizada.

El código EAN es la versión europea del UPC, y se creó en 1976. Es el estándar europeo que ha tenido aceptación a nivel mundial. Identifica a los productos comerciales en los supermercados y comercios minoristas con dígitos que indican país/empresa/producto con una clave única internacional. Más de 12.000 empresas en Argentina ya han codificado más de 350.000 productos.

La versión más difundida es el EAN-13 y consta de un código de 13 dígitos, uno más que el UPC. Los tres primeros dígitos indican el país (Argentina es 779), los seis siguientes a la empresa productora, y los tres posteriores al artículo o producto. Finalmente está el dígito verificador que le da seguridad al sistema. Para artículos de tamaño reducido se utiliza el código EAN-8.

UPC es una simbología de longitud fija, numérica, continua, y que emplea cuatro anchos distintos. El código UPC es un código denominado también (7,2) lo que significa que cada elemento tendrá un ancho de 7 módulos y tendrá dos barras y dos espacios. De aquí se deduce que la cantidad máxima de caracteres será 20 (como colocar cuatro elementos en sus distintos tamaños relativos en 7 posiciones). Se ve que la barra o espacio más ancho tendrá 4 módulos. Al ser un código numérico, UPC codifica los números del 0 al 9. Un dato interesante del código UPC-A es que, para facilitar la tarea del dispositivo de lectura, la altura de las barras debe ser mayor al ancho de cualquier mitad.

El valor nominal de X para todos los códigos UPC es de 13 mils. Se permite un factor de maximización de 0.8 hasta 2.0 lo que resulta en un rango de valores X imprimibles de 10.6 mils hasta 26 mils. [Pal91]

UPC se decodifica con algoritmos de lectura de la longitud de un eje al eje adyacente similar. Lamentablemente, la codificación de los dígitos 1 y 7 tienen similares distancias entre ejes (llamadas distancias T); un problema similar ocurre con las distancias T de los caracteres 2 y 8. Para evitar esta ambigüedad, el dispositivo lector UPC necesita obtener, en estos casos, los anchos de los elementos.

La simbología EAN tiene las mismas características estructurales que la simbología UPC, y es un superconjunto de UPC. Un dispositivo lector de EAN puede decodificar UPC pero no a la inversa. EAN tiene dos simbologías EAN-13 y EAN-8 que codifican 13 y 8 dígitos respectivamente. La EAN-13 contiene el mismo número de barras que UPC-A, pero codifica un decimotercer dígito en un patrón de paridad de los seis dígitos de la mitad izquierda. Este último dígito, en combinación con el doceavo, define dos caracteres especiales que representan el código de país. Para compatibilidad con UPC los caracteres 00, 01, 03, 04, 06 y 09 fueron asignados a los Estados Unidos. EAN-8 es utilizado para etiquetas de tamaño pequeño.

UPC/EAN han probado ser extremadamente exitosos en el mercado minorista (supermercados o comercios). En su uso normal, cada mensaje decodificado en un supermercado es verificado contra una base de datos antes de ser efectivamente utilizado. En este modelo se ha demostrado una alta confiabilidad y una gran seguridad de la información.

La simbología "Interleaved 2 of 5" es una simbología de alta densidad, con auto verificación, continua, numérica y que utiliza sólo dos anchos. Se ha utilizado mayormente en la industria de la distribución, y también en los depósitos. Cada elemento codifica en realidad dos dígitos, uno en las barras y uno en los espacios. Cada par de caracteres tiene 5 barras y 5 espacios, dos de las barras y dos de los espacios son anchos, y el resto son angostos. Por esto, la cantidad de caracteres decodificados siempre será par.

Los caracteres de arranque/parada consisten en dos barras angostas y dos espacios angostos. Esto hace que cualquier otro elemento de la simbología, que comience con esta misma distribución de barras y espacios, si el dispositivo lector examina el símbolo parcialmente, pueda ser interpretado como un caracter de parada; éste es un problema real aunque no muy frecuente de esta simbología. Para evitar este problema se trata de utilizar esta simbología para códigos de longitud fija donde el sistema que está detrás del dispositivo lector espere sólo recibir símbolos de una determinada longitud. Otra medida para garantizar la seguridad de la información es la de agregar un caracter de chequeo al final del símbolo.

Otra simbología muy conocida es Code 39. Se desarrolló en 1974 porque algunas industrias necesitaban codificar datos alfanuméricos. Actualmente es la más ampliamente difundida en el espacio que no ocupa UPC/EAN. Es una simbología discreta, tiene auto verificación, es de longitud variable, tiene sólo dos anchos, y puede ser impresa en una amplia variedad de tecnologías. Se la suele llamar también Código 3 de 9.

El juego de caracteres de esta simbología sólo tiene 43 elementos aunque el máximo posible de codificar es 128. Se utilizan 26 letras, 10 números y 7 caracteres especiales. Cada caracter tiene cinco barras y cuatro espacios, lo que hace un total de 9 elementos, tres de los cuales son anchos y seis angostos. Cada símbolo comienza y termina con un asterisco (*) que cumple la función de caracter de arranque/parada. Cada caracter está separado de su vecino por un espacio que no contiene información (por eso es discreta); debido a la similitud entre los caracteres asterisco (*) y la letra "P" (uno es espejo del otro), el ancho del Inter-Character Gap está limitado en tamaño para prevenir malas interpretaciones especialmente en casos de lecturas parciales. Esta simbología produce símbolos relativamente largos y si el espacio físico es un factor de consideración ésta puede no ser la mejor solución.

Otra simbología de interés el Code 128. Esta simbología fue creada en 1981 cuando fue necesario un mayor juego de caracteres que lo que puede proporcionar Code 39, y tiene una muy alta densidad de información. Es una simbología de longitud variable, es continua, tiene cuatro diferentes anchos para las barras, y justamente tiene 106 diferentes caracteres. Code 128 consta de 11 módulos, y cada caracter tiene tres barras y tres espacios.

Por último, Code 93 fue introducida en 1982 para complementar Code 39. Es una simbología de longitud variable, continua, que emplea cuatro anchos distintos. Cada caracter ocupa 9 módulos con tres barras y tres espacios. Las dos simbologías (Code 93 y Code 39) pueden ser mezcladas libremente en un símbolo, y utilizando cualquier tipo de dispositivo de lectura no es necesario hacer ningún cambio en el software de decodificación. Codifica 47 caracteres, y tiene iguales caracteres de arranque y parada, aunque al final del caracter de parada se agrega una barra para indicar el final del símbolo. Code 93 no tiene auto chequeo, pero ofrece una alta seguridad en la información a través del uso de dos caracteres de verificación. [Pal91]

Una medida de comparación entre los códigos es la densidad. Se puede observar una comparación en la Tabla 2-1, asumiendo que Code 39 tiene una relación ancho/angosto de 2.2 y que X = 10 mils.

Tabla 2-1. Longitud de símbolo.

Simbología	Mensaje = "TESTING 123"	Mensaje = "123456 ABC"
Code 39	1,76"	1,62"
Code 128	1,56"	1,12"
Code 93	1,36"	1,27"

Según la Tabla 2-1, se puede decir que Code 128 es la simbología de mayor densidad para información que contiene largas cadenas de números, y que Code 93 es la simbología de mayor densidad para información para información alfanumérica.

Existen también los denominados "Códigos de Barras de dos dimensiones". Una parte de éstos son los llamados Códigos Apilados porque corresponden a símbolos de una dimensión apilados uno encima del otro, formando filas de símbolos de una dimensión de poca altura, algunos ejemplos son Code 49 y Code 16K. Otros códigos son verdaderos códigos en dos dimensiones, como Vericode, Data Code, UPC Code y PDF417.

La elección de una simbología es a menudo muy sencilla ya que casi todas las industrias tienen su estándar. Si uno desea enviar productos a supermercados deberá utilizar UPC/EAN, si lo hará al Departamento de Defensa de los Estados Unidos utilizará Code 39. Si el producto es una etiqueta de correo normalmente utilizará una simbología denominada Postnet. Para aplicaciones internas existe una gran variedad de simbologías a utilizar pero mucho dependerá de factores como el área física para el símbolo, el tipo de información a ser codificada, el tipo de equipamiento de impresión y del tipo de dispositivo de lectura a ser utilizado.

Otras características de interés son las siguientes:

- un gran juego de caracteres, según se necesite codificar información numérica o alfanumérica
- la facilidad de decodificar con alta confiabilidad
- la posibilidad de decodificar utilizando dispositivos de lecturas baratos
- la facilidad de impresión debido a una simple estructura física y amplias tolerancias
- una alta seguridad de la información, aún si se requiere imprimir en los extremos de la tolerancia de impresión permitida
- una alta densidad de información dado un tamaño mínimo de símbolo

Desafortunadamente, algunas de las seis características tienden a ser mutuamente excluyentes. Sin embargo, la mayoría de las simbologías poseen las cinco primeras.

Los indicadores de las simbologías abarcan aquellos que miden eficiencia y densidad en una dimensión y en dos dimensiones. El contenido de la información de un mensaje es medido en bits. Se asume que todos los caracteres de cualquier simbología son igualmente probables. El contenido de información de un único caracter está definido como:

$$I = \log_2 C \qquad \text{[Pav90]}$$

dónde I es la cantidad de bits por caracter o Información, y C es el tamaño del conjunto de caracteres.

Como ejemplo, en la Tabla 2-2 se puede ver la cantidad de información para un mensaje de un caracter, en función del tamaño del juego de caracteres:

Tabla 2-2. Cantidad de Información según el Juego de Caracteres. [Pal91]

C	I
10 (Interleaved 2 of 5)	3,32
26	4,70
43 (Code 39)	5,43
47 (Code 93)	5,56
103 (Code 128, Code 16K)	6,69
2401 (Code 49 Word)	11,23

El contenido de información para un mensaje que contiene más de un caracter es igual a:

$$\log_2(C^N) \qquad \text{[Pav90]}$$

o

$$N \log_2 C \qquad \text{[Pav90]}$$

dónde C es el tamaño del conjunto de caracteres y N es el número de caracteres en el mensaje.

Cuando se comparan las eficiencias de las diferentes simbologías de una dimensión, es conveniente examinar la eficiencia lineal, que se define como:

$$eficienciaLineal = \frac{bitsDeDatos}{longitudEnMódulos} \qquad \text{[Pal91]}$$

Aquí, el término "módulos" se refiere al tamaño mínimo de elemento, o la dimensión X.

Tomando como ejemplo a Code 39; en el caso de un mensaje con un único símbolo contiene 5.43 bits de información. Si se imprime con una relación ancho/angosto de 2.5:1, cada caracter tendrá una longitud de 14.5 módulos de ancho (el Inter-Character Gap mide 1 módulo), y la longitud total de todo el símbolo será de 42.5 módulos. La eficiencia bruta es de 5.43/14.5 = 37.4%. Cuando se le agrega la sobrecarga (caracteres de arranque/parada, etc), la eficiencia neta es del 12.8%. Obviamente, la eficiencia neta se incrementa para mensajes más largos. La eficiencia lineal bruta de Interleaved 2 de 5 es de 41.5%, de Code 39 es de 37.4%, de Code 93 es de 61.7%, y de Code 128 es de 60.8%.

Como se ha visto, la eficiencia neta puede variar dependiendo de la longitud del mensaje. Se muestra en la Tabla 2-3 la eficiencia neta en ejemplos de 20 y 100 bits.

Tabla 2-3. Eficiencias Bruta y Neta para distintas simbologías. [Pal91]

Simbología	*Eficiencia Bruta Lineal*	*Eficiencia Neta Lineal con mensaje de 20 bits*	*Eficiencia Neta Lineal con mensaje de 100 bits*
Interl. 2 of 5 (impreso a 2.5:1)	41,5 %	3 caract.: 35,3%	15 caract.: 40,1%
Code 39 (impreso a 2.5:1)	37,4 %	4 caract.: 25,2%	19 caract.: 34,0%
Code 93	61,7 %	4 caract.: 30,0%	18 caract.: 50,2%
Code 128	60,8 %	3 caract.: 35,2%	15 caract.: 50,9%

Un código de barras también ocupa espacio en altura, que en el caso de simbologías de una dimensión no codifica información alguna. Las simbologías actuales utilizan símbolos suficientemente altos como para facilitar el proceso de escaneo con dispositivos manuales. El índice que se utiliza para medir la eficiencia en área es el siguiente:

$$eficienciaArea = \frac{bitsDeDatos}{areaEnMódulosCuadrados} \qquad \text{[Pal91]}$$

Volviendo al ejemplo de 100 bits (18 caracteres), en Code 93 el estándar requiere una altura de al menos el 15% de la longitud del símbolo [Pav90]. Según la propia estructura del símbolo, eso lleva un área de 5970 módulos cuadrados por lo que la eficiencia neta de área es del 1.7%.

A título informativo, otras simbologías de códigos de barras apilados como Code49 o Code16K, alcanzan eficiencias netas de área cercanas al 10% para mensajes de 200 bits.

En términos comparativos, si se analiza el área ocupada por distintas simbologías de interés para mensajes de aproximadamente 20 caracteres alfanuméricos (y en el caso de Code 39 una relación ancho/angosto de 2.5:1), los resultados son los mostrados en la Tabla 2-4:

Tabla 2-4. Area en módulos cuadrados para distintas simbologías. [Pal91]

Simbología	*Area en módulos cuadrados*
Code 39	Aprox. 15.200
Code 128	Aprox. 8.200
Code 93	Aprox. 7.100
Codablock, Code 16K, Code 49	Entre 2.500 y 4.500
PDF 417, Vericode, Data code	Entre 290 y 670

2.4 Estándares de Códigos de Barras

Existen tres tipos de estándares para Códigos de Barras: de simbología, de aplicación y de calidad de impresión.

Los estándares de simbología especifican la estructura del código de barras. Describen cuanta información es codificada en una disposición de barras y espacios de anchos variables que conforman un símbolo de código de barras. Además de describir la codificación, especifican otros parámetros adicionales que se muestran a continuación.

Las tolerancias de impresión describen cuan precisos deben ser controlados los anchos de las barras y espacios durante el proceso de impresión. Si esas tolerancias son excedidas, la *tasa de lectura en primera instancia* caerá y la *tasa de error de sustitución* podrá sufrir incrementos dramáticos. Originalmente la tolerancia de impresión fue calculada equilibrando con la tolerancia del algoritmo de decodificación del dispositivo de lectura con la del proceso de impresión. La *tasa de lectura en primera instancia* indica el porcentaje de éxito en las lecturas, y la *tasa de error de sustitución* muestra el porcentaje de sustituciones de caracteres ante una decodificación errónea del símbolo.

Las tolerancias de impresión varían con el valor de X. En simbologías de dos anchos, las tolerancias de impresión también varían en función de N, el número de caracteres del mensaje.

Debido a que los códigos de barras son una tecnología óptica, también existen propiedades ópticas. Un estándar de código de barras especifica la reflectividad mínima permitida de los espacios de los símbolos y, directa o indirectamente, la reflectividad máxima permitida por la barras. Todas las mediciones están hechas a una longitud de onda específica, usando una geometría particular de medición.

Los defectos localizados en el proceso de impresión pueden interferir negativamente en el proceso de decodificación. Las *manchas* ("spots" en inglés) son áreas oscuras no deseadas en los espacios de un símbolo. Los *blancos* ("void" en inglés) son manchas claras en las barras, y las *asperezas* en los ejes ("roughness" en inglés) son las partes desparejas en la frontera entre una barra y un espacio. Las asperezas pueden llevar a una determinación incorrecta de las fronteras entre las barras y los espacios. Manchas y blancos demasiado grandes pueden causar una sensación de la existencia de una barra o un espacio donde no lo hay. Para una decodificación correcta, al menos un haz de luz debe recorrer el símbolo de punta a punta sin encontrar ningún obstáculo.

Los estándares de simbología usualmente especifican un valor de X (el valor nominal del ancho de la barra o espacio más angosto) para asegurar compatibilidad entre el equipamiento de impresión y de lectura.

Un estándar de aplicación para código de barras da información específica sobre el uso de los códigos de barras en una aplicación o industria particular. Referencia un estándar de simbología, luego describe que información será codificada, cuál será etiquetada y donde será colocado el símbolo en el objeto. A menudo también especifica la orientación del símbolo, el formato de la etiqueta, la altura mínima de las barras, el rango válido de X, la longitud de onda, y detalles de contenidos de información de una aplicación. Existen muchos estándares de aplicación, la industria automotriz tiene varios, la asociación del aluminio hizo una especificación para aplicaciones especificando el uso de códigos de barras sobre productos de aluminio, la industria gráfica, la industria de la salud, entre otras.

Los estándares de simbología incluyen límites en las tolerancias dimensionales, manchas, blancos, asperezas, reflectividad y contraste. Si estos requerimientos son seguidos meticulosamente no habrá necesidad de tener un estándar de calidad de impresión por separado, la calidad será excelente y la performance del sistema no se verá comprometida. La calidad de impresión percibida por el dispositivo lector puede variar dependiendo de su resolución. Un escáner con más alta resolución puede localizar imperfecciones que pueden no ser visibles para otro de menor resolución. Debido a esto, usualmente los estándares de calidad de impresión especifican el tamaño y forma de la apertura de la medición usada para la evaluación. Muchas de las aplicaciones de códigos de barras usan símbolos que se desvían en algunos aspectos de los estándares, pero en ciertos contextos o ciertas aplicaciones, las mayores tasas de error de lectura y de sustitución son tolerables. Para cuantificar estos desvíos en una forma en la que sea predecible, el organismo de estándares de Estados Unidos ANSI desarrolló un documento sobre estándares de calidad de impresión de códigos de barras. En este documento el símbolo a ser evaluado está sujeto a un mínimo de 10 escaneos de evaluación usando una diámetro de foco del haz de luz de diámetro especificado. Los escaneos se hacen a espacios regulares, de abajo hacia arriba, a lo alto del símbolo. La señal reflejada de cada escaneo es evaluada y calificada contra criterios específicos de contraste, modulación, decodificabilidad, y defectos. A cada escaneo se le asigna una calificación que es la peor de todas las calificaciones obtenidas en cada criterio de evaluación. Luego, la calificación total es el promedio de las 10 calificaciones individuales obtenidas. Esta calificación tiene sentido sólo si se ha especificado el diámetro del foco del haz de luz y la longitud de onda para la medición. [Pal91]

2.5 Equipamiento de lectura

Un dispositivo de lectura de código de barras es un equipamiento utilizado para extraer la información que está codificada ópticamente en un símbolo de código de barras y la convierte en información digital procesable por una computadora. El lector realiza una digitalización de la información utilizando una fuente de luz reflejada en el símbolo, y luego la información se envía a la computadora como si se hubiese ingresado por teclado.

Para simbologías de una dimensión, un lector de código de barras tiene que realizar siete funciones básicas:

1. Encontrar los elementos correctos
2. Determinar los anchos de cada barra y espacio de cada símbolo
3. Cuantizar los anchos de los elementos en un número de niveles apropiados para la simbologías que está siendo usada
4. Asegurar que los anchos cuantizados son consistentes con todas las reglas de codificación para esa simbología
5. Si es necesario, revertir el orden de la información
6. Confirmar que las quiet zones existen a cada lado del símbolo y son válidas
7. Confirmar que cualquier caracter de chequeo sea consistente con la información decodificada

El segundo paso es realizado por un sistema de escaneo electro-óptico en combinación con software que está almacenado en el microprocesador del lector. Los cinco pasos siguientes son manejados por rutinas de software.

Un lector consta de dos partes, un dispositivo de entrada y un decodificador. Pueden estar físicamente juntos o separados. [Pal91]

Un símbolo de código de barras es iluminado por una fuente de luz visible o infrarroja, las barras oscuras absorben la luz, y las barras blancas la reflejan. El escáner recibe ese reflejo y transforma esas fluctuaciones de luz en impulsos eléctricos los cuales copian las barras y espacios. El decodificador utiliza algoritmos matemáticos para traducir esos impulsos eléctricos es un código binario y transmite el mensaje decodificado a un terminal manual o PC.

Los dispositivos de entrada usan diodos emisores de luz visible o infrarroja, lásers de Helio-Neón, diodos láser de estado sólido (también de luz visible o infrarroja), o bien una fuente de luz externa. Algunos de ellos necesitan estar en contacto con el símbolo, otros pueden leer desde una distancia de hasta algunos metros. Algunos son fijos, otros son móviles o portátiles.

Existen distintos tipos de lectores de códigos de barras. Uno de ellos es el lector tipo pluma o lápiz. Fue muy popular hace un tiempo debido a su bajo precio y funcionalidad. El operador debe colocar la punta del lector en la quiet zone que está al inicio del símbolo, y debe deslizar el lápiz a velocidad e inclinación constante hasta el otro extremo. Esto requiere cierta habilidad del operador y en ocasiones son necesarios varios escaneos hasta conseguir una lectura correcta. Son muy prácticos cuando se leen símbolos colocados sobre superficies duras, planas, y de preferencia horizontales, y funcionan muy bien en símbolos impresos en muy buena calidad. El tipo de dispositivo es resistente a golpes laterales aunque sobre la punta lo pueden deteriorar fuertemente.

Los lectores de ranura o slot son básicamente lectores tipo pluma montados sobre una caja. Allí la lectura se realiza al deslizar una tarjeta o documento por la ranura, estando el lector en una posición fija y sin movimiento. La probabilidad de lectura en una primera oportunidad es mucho mayor, pero el símbolo debe estar alineado apropiadamente y colocado cerca del borde de la tarjeta o documento. [Pal91] Un ejemplo de este tipo de lectores es el que está en la puerta de ingreso a los cajeros automáticos.

Los lectores de tipo rastrillo o CCD, son lectores de contacto que emplean un fotodetector CCS (Dispositivo de Carga Acloplada) formado por una fila de LEDs emisores de luz similar al encontrado en cámaras de vídeo. Se requiere hacer contacto físico con el símbolo pero a diferencia de las lectoras de tipo lápiz no hay movimiento que degrade la imagen al escanearla.

Los lectores CCD de proximidad utilizan un mecanismo completamente electrónico, como si se tomase una fotografía del código. Para utilizarlo no se requiere contacto con el símbolo pero debe hacerse la lectura a corta distancia. Este tipo de lector tiene problemas de lectura en superficies curvas o irregulares.

Los lectores láser de proximidad también requieren corta distancia de lectura pero tienen una performance superior que los CCD debido a su potente luz láser. Incluso, logra mejores resultados en superficies curvas e irregulares que sus competidores.

Los lectores láser tipo pistola utilizan un mecanismo activador para prevenir la lectura accidental de otros códigos de barras. Un espejo rotatorio u oscilatorio dentro del dispositivo mueve el haz de un lado a otro a través del código de barras, de modo que no se requiere movimiento por parte del operador. Por lo general pueden leer códigos estropeados o mal impresos, en superficies irregulares o de difícil acceso, como el interior de una caja. Son más resistentes y aptos para ambientes más hostiles. La distancia de lectura es de entre 2 y 20 cm del código pero algunos lectores más avanzados pueden hacerlo hasta 30 cm, 1.5 metros y 5 metros.

Los lectores láser fijos son básicamente los mismos que el tipo anterior, pero montados sobre una base. La ventana de lectura se coloca frente al código a leer (generalmente orientada hacia abajo) y la lectura se dispara al pasar el artículo que contiene el código frente al lector y activarse un sensor especial. Esta configuración se encuentra frecuentemente en bibliotecas ya que libera las manos del operador para que pueda pasar el libro frente al lector. También se utiliza en sistemas automáticos de fábricas y almacenes, donde el lector se coloca sobre una banda transportadora y lee el código de los artículos que pasan frente a él.

Los lectores láser fijos omnidireccionales se encuentran normalmente en las cajas registradoras de los supermercados. El haz de luz láser se hace pasar por una serie de espejos que generan un patrón omnidireccional, otorgando así la posibilidad de pasar el código en cualquier dirección. Los productos a leer se deben manipular y pasar a mano frente al lector. Son muy efectivos porque tienen una muy alta tasa de lectura en primera instancia. [Pal91]

Asimismo existen dispositivos llamados lectores autónomos que se utilizan en línea de producción o en procesos automatizados o de cinta transportadora. Las tecnologías utilizadas varían según lo requiera la aplicación.

Es significativo destacar la importancia que tiene la resolución del dispositivo de lectura. La barra o espacio que un dispositivo de lectura puede resolver es una función del tamaño del foco del haz de luz, ya sea su diámetro si se trata de un haz circular, o de su ancho si el haz es asimétrico. El tamaño del foco del haz de luz debe idealmente ser no más ancho que el ancho del elemento más angosto del símbolo, aunque algunos dispositivos permiten usar un tamaño más grande.

Si el tamaño del foco del haz de luz es más pequeño que el ancho del elemento más angosto del símbolo, el lector podría interpretar incorrectamente que ciertas imperfecciones como manchas o blancos como espacios o barras. Para minimizar este problema es deseable que el tamaño del foco del haz de luz coincida con el ancho del elemento más angosto del símbolo. Un problema muy común es que algunas aplicaciones utilizan un amplio rango de densidades de símbolos por lo que el tamaño del foco del haz de luz no puede ir bien en todos los casos y puede funcionar mal para casos donde X tienda a ser grande. Lo que se suele hacer es utilizar un foco del haz de luz en forma oval que esté orientado en el sentido de las barras, y de ésta manera se evita el reconocimiento incorrecto de barras y espacios que en realidad son defectos de impresión.

El decodificador forma parte del equipamiento de lectura, y es la parte que analiza la señal producida por el dispositivo de entrada, y descifra la información codificada en el símbolo de código de barras. La información resultante es transmitida luego a una computadora.

Un decodificador realiza los siguientes pasos:

1. Determinar si el dispositivo de lectura se encuentra sobre una barra o espacio. Esto se realiza normalmente comparando el valor de entrada con el umbral predeterminado
2. Medir en ancho de cada elemento que el dispositivo de entrada está escaneando (en el caso de un código que funciona mejor con técnicas de eje a eje similar, calcular las distancias T). Excepto los dispositivos de entrada CCD, no es posible la medición efectiva del tamaño físico de cada elemento. En su lugar, el decodificador mide el tiempo que le toma al dispositivo de lectura atravesar el elemento en el símbolo
3. Cuantizar los anchos de los elementos del símbolo (o distancias T)

4. Decodificar los caracteres de información codificados en el símbolo comparando el valor cuantizado con la tabla de valores válidos para cada caracter en el juego de caracteres
5. Si fuese necesario, dar vuelta el orden de los caracteres decodificados
6. Realizar chequeos adicionales para confirmar la validez de la información decodificada. Algunos de estos chequeos pueden ser, confirmar la validez de las quiet zones, chequear que el caracter de chequeo sea correcto, percibir que la velocidad del escaneo esté dentro de ciertos límites, percibir que una eventual aceleración no exceda el valor máximo predeterminado, realizar chequeos adicionales a la longitud del mensaje, espacios entre caracteres, caracteres de chequeo, y otros factores que pueden ser utilizados para maximizar la seguridad de la información
7. Transmitir la información decodificada a la siguiente etapa funcional del decodificador

Existen tres tipos de decodificadores, on-line, portables, y on-line portables. Los primeros están conectados a la energía eléctrica y a las computadoras en forma permanente, y transmiten la información decodificada directamente. Normalmente, estos decodificadores están instalados en equipos fijos donde el símbolo del código de barras es traído hacia el lector.

Los dispositivos portables contienen un dispositivo de almacenamiento local, y una batería. La información es retenida en el dispositivo para luego ser descargada toda de una vez a la computadora. Normalmente, el dispositivo es llevado hacia el símbolo para que éste sea escaneado.

Los dispositivos on-line portables son el mismo tipo de equipamiento con el agregado que se comunican con la computadora a través de señales de radio frecuencia. Del lado de la computadora, utiliza un controlador que puede comunicarse con varios dispositivos simultáneamente por radio frecuencia. El dispositivo utiliza una batería que mantiene el equipamiento de lectura y el de emisión encendidos. Lo mejor de este tipo de dispositivos es que pueden acercarse al símbolo sin sacrificar interactividad on-line.

2.6 Dispositivos de impresión

Existe una clasificación que divide a la impresión de códigos de barras en dos tipos: impresión fuera del sitio, e impresión en el sitio.

La impresión fuera de sitio se refiere a las tecnologías que son utilizadas para reproducir símbolos de códigos de barras para uso subsecuente. La producción es usualmente realizada en un lugar diferente a donde los símbolos serán utilizados y a menudo la generación del símbolo es contratada a un tercero. Normalmente se utiliza la impresión fuera del sitio cuando se requiere la impresión de grandes volúmenes de símbolos idénticos. Debido a que existe una separación de tiempo entre la producción de los símbolos y su utilización efectiva a este proceso se lo describe como un proceso por lotes (batch).

La impresión en el sitio se utiliza para crear símbolos de códigos de barras al mismo tiempo que van siendo utilizados. La información codificada en cada símbolo puede ser diferente y es ingresada vía teclado local o por una computadora. Se conoce a ésta técnica de impresión en sitio como impresión a demanda. Obviamente también se puede utilizar la impresión en sitio para realizar grandes volúmenes de símbolos idénticos.

Los símbolos de códigos de barras pueden imprimirse en una amplia variedad de sustratos: rótulos, etiquetas, páginas, formularios, paquetes convencionales, etc.

Normalmente, los símbolos de códigos de barras son impresos junto con otra información útil al ojo humano.

Evidentemente, una combinación de impresiones fuera del sitio y en el sitio también puede ser utilizada. Por ejemplo, cierta información que no cambia puede estar preimpresa fuera del sitio, y la información variable puede incorporarse con una impresión en el sitio.

Existen técnicas de impresión fuera del sitio con tinta húmeda que incluye impresión directa en papel, litografía offset, flexografía, impresión en rotor, y la rueda de tinta. Otras técnicas son la fotocomposición, el estampado en caliente, y fijación láser. Las técnicas de tinta húmeda son las más tradicionales, y todas coinciden en aplicar tinta sobre un sustrato. La diferencia radica en la forma en la que transfieren la tinta al sustrato. En casi todos los casos se utiliza una imagen positiva o negativa para generar las láminas o chapas para impresión. Estas láminas son llamadas maáters. El tamaño de ancho de las barras y espacios impresos difiere de los del máster dependiendo del proceso de impresión y de las características del sustrato y la tinta. Esta desviación ya es conocida por lo cual el máster es ajustado para que la impresión final sea correcta. Desafortunadamente, la reducción del tamaño de ancho de las barras varía según el proceso, el sustrato y las tintas.

La técnica de impresión directa en papel permite un valor de X tan pequeño como 8 mils que es bastante bueno y tiene una buena calidad de impresión; al utilizar ruedas, puede utilizarse para realizar impresiones con datos que varían secuencialmente desde una impresión a la siguiente. La técnica de offset también permite un valor de X tan pequeño como 8 mils, y también imprime con buena calidad de impresión; el único comentario es que requiere tintas basadas en aceite. La flexografía permite un X de hasta 10 mils y puede ser impreso en una amplia variedad de sustratos. La impresión en rotor permite un X de 12 mils, es muy barata, pero la calidad es mediocre. La impresión por ruedas de tinta tiene la desventaja que para tener un símbolo confiable el valor de X más chico no debe bajar de los 40 mils; la ventaja es que el equipamiento es barato.

A pesar que las técnicas de tinta húmeda son las más comunes, existen otras técnicas de impresión fuera del sitio, y también son utilizadas. La fotocomposición, por ejemplo, puede producir símbolos de código de barras de altísima calidad en una amplia variedad de sustratos fotosensitivos; tiene la mejor ventaja que es la de poder producir símbolos con un valor de X tan pequeños como 3 mils y con una excelente calidad incluso para información variable, pero la gran desventaja es que es extremadamente caro. En el estampado en caliente, la lámina metálica es calentada y luego presionada contra una cinta sensible al calor. La calidad de impresión es buena, pero el costo por símbolo es también muy caro.

Por determinadas razones, quizás comodidad, quizás flexibilidad, es que también son utilizadas las técnicas de impresión en el sitio. Las distintas tecnologías para este caso son: impresión de impacto de matriz de puntos, por tambor, térmicas, transferencia térmica, inyección de tinta, xerografía, magnetografía, y electrostática.

La impresión de impacto de matriz de puntos es aquella que contiene un martillo con agujas, que presionando sobre la cinta marcan los puntos sobre el papel. Para producir un símbolo de código de barras, una impresora de matriz de puntos sobreescribe puntos adyacentes para producir una aproximación a los ejes rectos de las barras. Cuando la cinta es nueva y tiene tinta el diámetro de los puntos impactados por las agujas es grande y por lo tanto la sobreescritura para producir la aproximación de puntos es buena, pero cuando la cinta se va quedando sin tinta

el diámetro de los puntos impactados es menor y el solapamiento de los puntos impactados es más pobre, reduciendo el ancho efectivo de las barras. La mayoría de las impresoras de matriz de puntos generan una dimensión X de entre 15 y 20 mils, produciendo una densidad de Code 39 de alrededor de 4 caracteres por pulgada. Para utilizar estas impresoras es necesario contar con un mantenimiento adecuado y tener mucho cuidado al seleccionar la cinta y el papel.

Las impresoras de tambores fueron diseñadas explícitamente para imprimir rótulos y etiquetas de código de barras. Una impresora de este tipo usa un tambor en continua rotación que tiene caracteres de código de barras y caracteres entendibles por humanos delineados en la superficie externa. Cuando un caracter a ser impreso rota dentro de la zona de impresión, un martillo dirige el sustrato (papel, vinilo o poliéster) dentro de una cinta y luego dentro del caracter delineado en el tambor. Tiene una excelente resolución en los ejes lo cual permite que el valor de X sea de 5 mils o quizás menos. La densidad en Code 39 es de 15 caracteres por pulgada. Por esto, este tipo de impresoras son muy utilizadas para etiquetar objetos pequeños.

Las impresoras térmicas se comenzaron a utilizar para códigos de barras recién en el año 1981, produciendo símbolos de alta calidad. Coincidió con el desarrollo de sustratos sensibles a la temperatura que darían imágenes estéticas con una aceptable vida útil. El principio básico de estas impresoras es que un sustrato de color claro es impregnado con una cobertura clara que cambia de color al negro o azul cuando es expuesta al calor excesivo (200 °C) por un período de tiempo (milisegundos). La imagen se forma por una reacción química en la cobertura sensible al calor, en los puntos en los que se le aplica temperatura. Cada punto calentado es de forma rectangular lo que facilita la impresión de códigos de barras. Los cabezales de impresión térmica tienen un alto desgaste y deben ser considerados un bien consumible. El tamaño de cada punto rectangular es de aproximadamente 10 mils cuadradas. Por otra parte, la velocidad de impresión es de entre 0.5 y 5.0 pulgadas por segundo, incluso más. Las grandes ventajas son la flexibilidad, la calidad de la imagen, la velocidad y el costo. Las únicas desventajas son que las etiquetas no deben estar expuestas a temperaturas ambiente que excedan los 60°C ni que permanezcan expuestas por más de dos días a la luz ultravioleta; por estas razones las etiquetas térmicas están usualmente restringidas a aplicaciones internas, por ejemplo dentro de una oficina, y allí son muy populares.

Las impresoras de transferencia térmica tienen muchas similitudes con las impresoras térmicas directas, la diferencia más notoria es que el cabezal térmico está en contacto con una goma especial que libera un pigmento por encima de una temperatura; ésta goma es quién está finalmente en contacto con el sustrato. La imagen resultante es estable y no está afectada por exposición a la temperatura excesiva o la luz ultravioleta. Son tan versátiles como las de impresión térmica directa con la ventaja adicional que el papel impreso es común, y el sistema puede ser utilizado en aplicaciones al aire libre. La única desventaja es la reposición de la goma especial.

Las impresoras xerográficas están basadas en una tecnología de utilización de papel común. Es la tecnología utilizada en las impresoras láser actuales. Se basa en que ciertas cargas eléctricas generan una imagen electrostática en el papel, que luego es puesto en contacto con el tóner cuyas partículas se ven luego atraídas por la imagen electrostática. La resolución de impresión es de 240 o 300 puntos por pulgada lineal. Las barras más angostas son de 3 puntos de ancho y las anchas están formadas por ocho o nueve puntos.

Las impresoras electrostáticas utilizan la técnica de componer una imagen electrostática en el papel, luego atraer las partículas de tóner líquido o en polvo, y finalmente fijarlas al sustrato a través de calor o presión. No utiliza ningún sistema óptico. Puede imprimir hasta 100 puntos por pulgada.

Las impresoras de deposición de iones son una mezcla entre las electrostáticas y las xerográficas. No utiliza ningún sistema óptico, pero la impresora utiliza un tambor de impresión y usa papel común. La técnica es generar una deposición de iones en el tambor formando la imagen, y luego atraer el tóner al papel para finalmente fundirlo al mismo con presión. Debido al uso de presión para la fusión, no deben utilizarse materiales de etiquetas laminadas. Pueden imprimirse hasta 150 puntos por pulgada.

Las impresoras magnetográficas son similares a las de deposición de iones, con la diferencia que en lugar de crear imágenes electrostáticas en el tambor se utilizan imágenes magnéticas. En algunas ocasiones éste proceso es denominado magnetolitografía. Se pueden imprimir hasta 100 puntos por pulgada. [Pal91]

2.7 Especificaciones de impresión y calidad del símbolo

Cada simbología tiene asociado un conjunto de especificaciones que deben ser respetadas durante el proceso de impresión. Cualquier desviación puede resultar en una caída en la tasa de lectura en primera instancia y un incremento en la tasa de error de sustitución. Las especificaciones de impresión incluyen límites en las tolerancias dimensionales, manchas, blancos, asperezas en los ejes, y reflectividad y contraste.

Las tolerancias en los anchos de los elementos impresos son distintas para cada simbología, pero todas están en función del valor de X.

Existen tres tipos de tolerancias, y son las siguientes:

1. Tolerancia de barra o espacio: es la desviación (positiva o negativa) desde el ancho nominal de cualquier elemento
2. Tolerancia eje a eje similar: es para simbologías continuas, y es la desviación desde el valor nominal entre ejes similares de elementos adyacentes dentro de un caracter
3. Tolerancia de caracter: es para simbologías continuas, y es la desviación al valor nominal de la distancia entre un eje principal de un caracter y el eje principal del caracter adyacente.

Las tolerancias en la impresión para una simbología dada están basadas asumiendo un cierto algoritmo para la determinación de anchos específico en el dispositivo lector. Esto es, se está asumiendo que el dispositivo lector también tendrá que tolerar algunas desviaciones. Existen algunos dispositivos verificadores que revisan si las dimensiones de un símbolo están dentro de la especificación. [Pal91]

Un símbolo cuyas dimensiones están dentro de la especificación puede todavía ser no legible si se encuentran excesivas "manchas", "blancos" o "asperezas en los ejes". Como ya se mencionó anteriormente, esto se agrava si el dispositivo de lectura tiene un tamaño del foco del haz de luz mucho menor que el valor de X utilizado, en cuyo caso una lectura podría interpretar algunos de estos defectos como barras o espacios adicionales, con los consiguientes dramáticos cambios en las tasas de primera lectura y de error de sustitución.

Un trabajo realizado por la ANSI cuantificó la cantidad de defectos de impresión permitidos. Los resultados dicen que están permitidos cualquier número de manchas y blancos mientras que se verifique lo siguiente para cada defecto:

1. No cubrirá más del 25% del área de un círculo cuyo diámetro es 0.8X
2. No cubrirá completamente un círculo cuyo diámetro es 0.4X

Las "asperezas" son una medida de la desviación del eje del elemento respecto a su posición vertical perfecta. Son percibidas como una diferencia en la medición del ancho de un elemento. Las especificaciones usualmente no indican nada especial acerca de las "asperezas", normalmente quedan especificadas en forma indirecta a través de la definición de la ubicación de los ejes.

Los códigos de barras son una tecnología óptica. La imagen impresa debe procesar ciertas propiedades ópticas si se utilizará un equipamiento de lectura. Todas las mediciones, por supuesto, deberán ser realizadas a una cierta longitud de onda.

Los dispositivos de entrada de códigos de barras responden a la diferencia en la reflectividad entre barras y espacios de un símbolo. Los dispositivos de lectura de códigos de barras idealmente tienen un diámetro de apertura del foco de 0.8X, pero en la realidad las propiedades de reflectividad de los símbolos leídos son bastante diferentes. Los sistemas de lectura de códigos de barras requieren un mínimo de diferencia en la reflexión de barras y espacios adyacentes para poder funcionar; éste mínimo varía según el equipamiento, sin embargo los fabricantes aseguran que existe un margen de seguridad entre la diferencia mínima entre barras y espacios respecto a lo especificado en el estándar de código de barras.

Usualmente los símbolos están especificados con dos parámetros, reflectividad del fondo y señal de contraste de impresión (PCS). La reflectividad del fondo es una medida de la reflectividad en los espacios del símbolo, y PCS especifica indirectamente el índice de reflectividad entre las barras y los espacios. Los valores de especificación para símbolos donde X es menor que 40 mils es de una reflectividad de fondo mínima de 50%, y y PCS mínimo de 75%. Este 75% de PCS indica que la reflectividad de las barras sea no más de un cuarto de la reflectividad de los espacios, pero incorporando el 50% de reflectividad de fondo mínima, la diferencia en reflectividad entre las barras y los espacios resulta en un mínimo del 37.5%. Para evitar ciertos casos en los que la diferencia en reflectividad es mucho mayor que el 37.5% (y por ende mucho mejor) pero donde alguno de los dos componentes que lo forman tiene valores que no coinciden con la especificación, en 1985 se introdujo un indicador de la mínima diferencia de reflectividad (MRD) que mide la diferencia en reflectividad entre la barra más clara y el espacio más oscuro utilizando un tamaño de foco del haz de luz de 0.8X. El MRD se define como un mínimo de 37.5% para símbolos con X < 40 mils y de un mínimo de 20% si X >= 40 mils. [Pal91]

2.8 Seguridad de la información

Un sistema de códigos de barras es exitoso si posee características de rapidez, economicidad y seguridad, pero también debe poseer características de seguridad como por ejemplo una tasa de error especificable. La tasa de error depende de si la simbología utilizada es segura, la calidad de la impresión es alta, y el equipamiento de lectura es apropiado. Asimismo, la tasa de error puede reducirse utilizando mecanismos de redundancia y chequeo de caracteres en los mensajes.

Todas las simbologías utilizadas son básicamente seguras, sin embargo pueden aparecer errores cuando se combinan algunas de las siguientes características:

1. Los anchos de los elementos se desvían del valor nominal
2. Hay excesiva reflectividad en las barras o baja reflectividad en los espacios lo cual lleva una señal óptica inadecuada al dispositivo de lectura
3. La presencia de "manchas" o "blancos" no deseados en el símbolo impreso
4. Los sistemas de lecturas no miden directamente los anchos de los elementos impresos; en su lugar, se mide el tiempo que tarda el foco del haz de luz en atravesar el elemento; también se toma en cuenta para esta medición la eventual aceleración del haz de luz
5. Las lecturas deben moverse completamente desde una quiet zone a la otra; las lecturas parciales a menudo ocurren, y eventualmente resultan en información decodificada errónea.

El grado de resistencia a uno o más de estos defectos varía según la simbología.

Se han hecho varios intentos por medir la resistencia teórica de las simbologías, sin embargo, aún es un trabajo inconcluso. Se ha podido realizar un análisis calculando el factor de una simbología respecto a tres tipos de errores de impresión:

1. Dos anchos de elementos con una desviación de caracter respecto al tamaño nominal de sus anchos, en el peor caso (un caracter ensanchándose y el otro encogiéndose)
2. Un ancho de elemento con una desviación de su valor nominal
3. Todas las barras con un ensanchamiento o encogimiento uniforme

El factor de seguridad indica, bajo condiciones de velocidad uniforme de escaneo, el porcentaje de la tolerancia total a fallas remanente luego que la tolerancia de impresión ha sido calculada. Por ejemplo, sin una simbología tiene un factor de seguridad del 65% significa que el 35% de la tolerancia se ha consumido en el proceso de impresión, y que el 65% restante se reserva para el deterioro potencial del símbolo impreso (se considerará marginal) y para el sistema de lectura.

Este factor de seguridad es necesario para compensar fallas en el dispositivo de lectura, como por ejemplo, efectos de aceleración de la lectura, falta de coincidencia en la resolución de lectura, errores de cuantización, y errores de digitalización.

Los valores teóricos de factor de seguridad para el peor caso son los siguientes: Code 39: 5/9, Interleaved 2 of 5: 56%, UPC-A: 32%, y Code 128: 32%. Luego, entre 1986 y 1987 la asociación AIM (Automatic Identification Manufacturers) auspició una serie de tests de performance sobre 7 distintos tipos de simbologías en la Universidad Estatal de New York. Las pruebas fueron exhaustivas e indicaron que UPC-E y Codabar son las simbologías con mayor tasa de error, y que Code 128 y Code 93 tienen la más alta seguridad de los datos. [Pal91]

Como ya se ha mencionado, la performance de un sistema de códigos de barras es altamente dependiente de la calidad del símbolo impreso. En una lectura de un símbolo pueden pasar tres cosas: que el lector decodifique correctamente la información, que el lector no decodifique nada, que el lector haya decodificado pero que la información no sea correcta. Para el primer caso se está haciendo mención a la tasa de lectura en primera instancia (FRR). Cuando un símbolo es de buena calidad y el dispositivo lector utilizado por personal entrenado, el FRR puede exceder el 90%. De todas maneras, los fabricantes deben asegurar un buen FRR cuando se leen símbolos que están dentro de la especificación, a través de un amplio rango de calidad de impresión. En el tercer caso se está haciendo mención a

la tasa de error de sustitución (SER). Normalmente, ambas tasas son altamente dependientes entre sí. [Pal91]

Como en cualquier esquema de transmisión de información, la tasa de error de un sistema de códigos de barras puede ser drásticamente reducida utilizando caracteres de chequeo. Un caracter de chequeo es un caracter adicional que está basado en un cálculo matemático respecto de algunos de los caracteres con información; el decodificador, luego, confirma que el caracter de chequeo observado es consistente con el resto de los caracteres observada antes de aceptar la información.

Cuando se utilizan caracteres de chequeo, la mejora en la seguridad de la información es muy significativa. En Code 39, por ejemplo, la tasa de error se reduce en un factor de 43%; en la práctica, incluso, es mayor que esto. Code 93, Code 128, Code 49 y Code 16K, entre otros, incorporan caracteres de chequeo en su simbología; las tasas de error para ellas han probado ser extremadamente pequeñas. [Pal91]

3. *Introducción a los DataGlyphs*

3.1 Definiciones y Descripción General

Desarrollados por Xerox PARC (Palo Alto Research Center), los *DataGlyphs* constituyen una tecnología que permite embeber una gran cantidad de información digital en una excelente interacción, con los dibujos, con los textos y con documentos que contienen ilustraciones en color o blanco y negro.

Un DataGlyph es un reticulado o tablero compuesto por elementos o marcas llamadas glyphs. Un *glyph* es una línea diagonal a 45°, de un mínimo de 3 pixeles de longitud, que representa un cero o un uno según esté recostada hacia la izquierda o hacia la derecha. En ese reticulado todas las celdas virtuales tienen un determinado tamaño donde allí dentro hay un glyph. Para un glyph de la longitud descripta, la celda puede ser de 5 pixeles de lado.

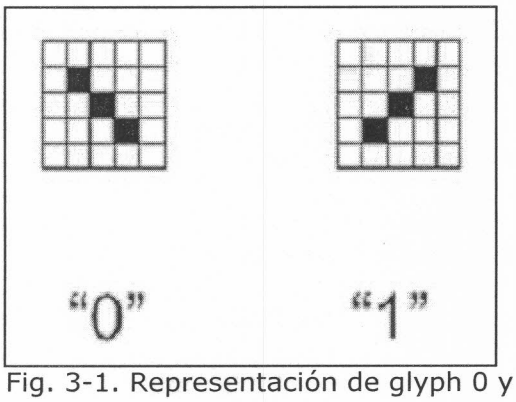


Fig. 3-1. Representación de glyph 0 y 1.

El Dataglyph constituye también un código con auto seguimiento. La propiedad de auto seguimiento está indicada en la existencia de glyphs en posiciones específicas; la presencia del glyph es el mecanismo de auto seguimiento, y una o más de sus características, como la orientación del glyph, codifica la información.

Dentro del DataGlyph es muy sencillo codificar estructuras lógicas. La disposición del código de DataGlyphs reserva un espacio para el enmarcado de un bloque de glyphs, que se utiliza a su vez como mecanismo de sincronización. La sincronización permite una lectura ordenada de los bits aún cuando exista una sustancial distorsión o daño en la imagen. En el caso original, los DataGlyphs reservan una de cada quince filas y una de cada quince columnas para enmarcado y sincronización.

El hecho que los glyphs sean líneas a 45° no es casual. Un aspecto clave de la tecnología de los DataGlyphs es su carácter estético y gráfico. Cuando todos los glyphs están colocados en la estructura, el código tiene una textura homogénea ya que los elementos tienen el mismo número de pixeles oscuros y la apariencia visual es sustancialmente independiente de la inclinación de los elementos. Se ha probado que en esta forma las líneas son visualmente no obstructivas por lo que pueden componerse imágenes, logos, o dibujos con glyphs. A una distancia prudencial el ojo humano no detectará esta composición subyacente, tan sólo distinguirá la imagen, el logo o el dibujo que se compuso. A su vez, los glyphs pueden variarse en tamaño, forma y color, de manera que el potencial de esta tecnología es enorme. La técnica de componer una imagen utilizando DataGlyphs se denomina *GlyphTones*. (Ver Fig 3-2)

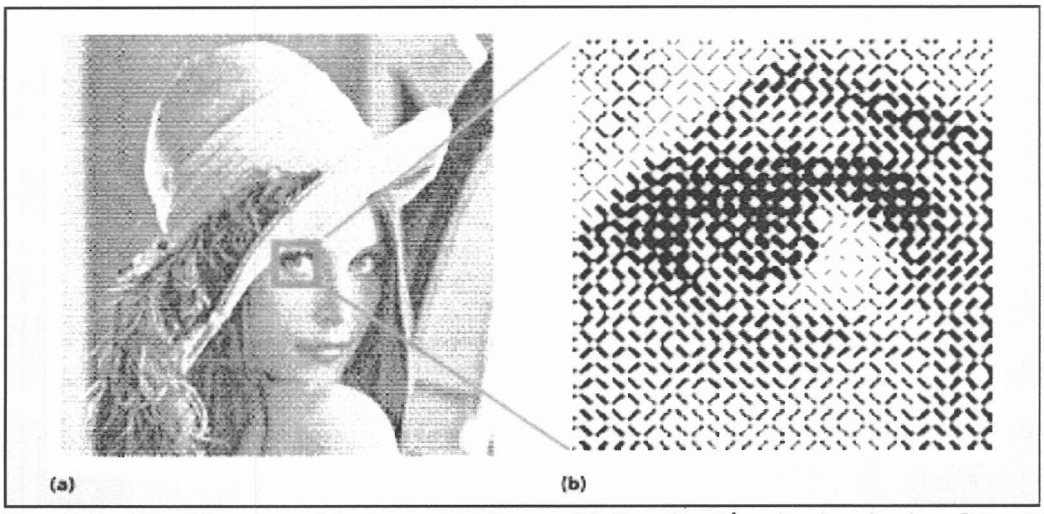


Fig. 3-2. Glyphtones. a) Imagen de Lena, b) Ampliación de de glyphs. [Hec01]

A diferencia de muchos códigos de barras, los Dataglyphs son flexibles en forma y tamaño. Pueden ser utilizados en superficies como papel, etiquetas, plásticos, espejos o metales. Su estructura y su robusto esquema de corrección de errores los hacen ideales para su utilización en superficies curvas y otras situaciones donde los códigos de barras normalmente fallan.

Se han realizado implementaciones en el mercado de las partes de aviones, en el segmento de servicios financieros, en el de software, en el de gobierno, en el de salud, y en el sector farmacéutico. Las aplicaciones incluyen la administración de la documentación, la prevención del fraude, el seguimiento de inventarios, las tarjetas de identificación, el marcado de partes, y el etiquetado de productos. [WWW1] [Hec01]

3.2 Datos Técnicos

Los Dataglyphs codifican información –texto, datos, gráficos- en miles de pequeños glyphs. Cada glpyh es una línea diagonal a 45° tan corta como una centésima (1/100) de pulgada o menos, dependiendo de la resolución de impresión y lectura (escaneo) que se utilice.

Como ya se vio la información es agrupada en bloques de unas pocas docenas de bytes cada uno, a los cuales se les adiciona información para la corrección de errores. Cada aplicación individual determina la cantidad necesaria de información para la corrección de errores que requiere. Por supuesto, mayores niveles de información para la corrección de errores requieren Dataglyphs más grandes para una cierta cantidad de información pero mejoran la confiabilidad con la cual la información puede ser leída. Habrá que observar el entorno en el cual se utilizará el Dataglyph, por ejemplo, si se trata de una transmisión de fax o si el sustrato sobre el que estará el Dataglyph está sujeto a mucha manipulación habrá mucho ruido y habrá que agregar información para la corrección de errores.

Luego de haber codificado y agregado la información para la corrección de errores, los bytes de información se dispersan aleatoriamente a través del área del Dataglyph. Así, si cualquier parte del DataGlyph se daña severamente, el daño a un bloque individual de información será ligero, y el código de corrección de errores podrá fácilmente recuperar la información codificada, a pesar del daño.

Las características de existencia de información para la corrección de errores y la dispersión al azar de la información le da a los Dataglyphs un muy alto nivel de confiabilidad.

La cantidad de información que puede ser codificada en un DataGlyph de un tamaño dado varía con la calidad de la impresión y el equipamiento de lectura utilizado. Por ejemplo, con un código de barras de una y dos dimensiones, el mínimo tamaño que puede ser usado es de 0.0075 pulgadas, 3 puntos a 400 dpi. A esa densidad, con una altura mínima, Code 39, el código de una dimensión de propósito general más ampliamente utilizado, puede sólo lograr una densidad de aproximadamente 25 bytes binarios por pulgada cuadrada. Code 128 puede lograr 40 bytes por pulgada cuadrada.

Los códigos de dos dimensiones como PDF417 lo hacen mejor, pueden lograr una máxima densidad aproximada de 370 bytes (2960 bits) por pulgada cuadrada sin ningún tipo de información de corrección de errores, a 400 dpi. Pero si se agrega una cantidad realista de información para la corrección de errores, un 27%, la efectividad real es de 270 bytes por pulgada cuadrada.

A la misma resolución (400 dpi) y cantidad de información para la corrección de errores (27%), DataGlyphs puede lograr casi 500 bytes por pulgada cuadrada.

De todas formas, la densidad de los DataGlyphs está determinada por cuatro factores:

1. La resolución a la cual la codificación es creada y leída. Los dispositivos de alta resolución como impresoras láser de oficina y dispositivos lectores de documentos permiten patrones de marcado más densos, y así una codificación más densa, mucho más que con dispositivos de baja resolución como impresoras de matriz de puntos o faxes
2. La cantidad de información para la corrección de errores. El proceso de impresión y lectura inevitablemente degrada la información codificada en la imagen. En codificación de alta densidad, donde los defectos de la impresora y el dispositivo de lectura son importantes comparados con el tamaño de la codificación, muchas de las características de codificación se perderán o serán mal leídas como resultado de esa degradación. Como contrapartida, algunos sistemas de redundancia deben ser usados para sostener la tasa de errores en límites razonables, esto es, para corrección de errores. La codificación redundante consume espacio extra y reduce la densidad efectiva de la información. Como ejemplo, el tamaño del DataGlyph que se requiere para codificar 80 bytes de información a ser transmitidos por fax es enormemente desproporcionado respecto al mismo DataGlyph impreso en una impresora de alta calidad, debido a la cantidad de información para la corrección de errores necesaria para dar confiabilidad a un DataGlyph enviado a través de ese medio. Como se ve, habrá que calcular cuanta información para la corrección de errores será necesaria en cada caso, y esto dependerá de la aplicación puntual. Las densidades de información medidas sin la incorporación de información para la corrección de errores representan límites teóricos más que de uso práctico
3. La compresión de información utilizada. La información puede ser comprimida a medida que se va codificando. Por ejemplo, si toda la información es numérica, no habrá necesidad de usar un byte (8 bits) por dígito, 3,32 bits será suficiente. Cuando el texto sea codificado, puede ser comprimido en escala de dos o más según la técnica de compresión utilizada
4. La sobrecarga fija de la sincronización. Para DataGlyphs el enmarcado de la sincronización es una proporción fija del área de información. Los DataGlyphs también tienen un encabezado fijo muy pequeño

Existe una especificación técnica propietaria de Xerox denominada *DG500*, para desarrolladores o implementadores, donde se describen los DataGlyphs

Al ser los DataGlyphs una tecnología propietaria de Xerox PARC, no se ha tenido acceso a información necesaria para conocer el funcionamiento interno de la misma. Toda lo expuesto es información pública. Para el presente trabajo, por lo tanto, se han utilizado los DataGlyphs a través de librerías para desarrolladores puestas a disposición del público por Xerox PARC en su sitio de Internet. [WWW1] [Hec01]

3.3 Descripción de las librerías de Xerox

Se han utilizado para el desarrollo de este trabajo 2 funciones de la librería de Xerox: *dgencode* y *dgdecode.*

dgencode se utiliza para generar un Dataglyph de N x N glyphs, a partir de un texto de entrada. *dgdecode* se utiliza para decodificar dicha matriz, generando el texto usado como parámetro de entrada del *dgencode*.

Ambas funciones llevan una serie de parámetros, además del texto de entrada o la matriz, según corresponda. Si bien en el trabajo se utilizaron los valores por defecto, se enunciarán cuales son dichos parámetros.

Función *dgencode*

Modo de uso: dgencode [opciones] [archivo de entrada]

Opciones:

Help	Muestra un resumen de los comandos.
Id <string>	Licencia requerida para activar el Toolkit de Dataglyph.
Cell <int>	Selecciona el espacio entre marcas de glyph, en pixeles. El valor por defecto es 7.
Mark <int>	Selecciona el tamaño de la marca de glyph. Se especifica N, donde la marca de glyph debe entrar en un cuadrado de (N-2)x(N-2) en pixeles. El valor por defecto es igual al espacio entre marcas de glyph (ver *-cell*. Notar que esta opción es ignorada cuando *-image* está presente, entonces el algoritmo de modulación dictará el tamaño de las marcas.
Weight <int>	Selecciona el peso de las marcas de glyph. A mayor peso, mayor tamaño tendrá la marca de glyph en pixeles; el ancho de la marca de glyph permanece constante cuando el peso está por debajo de cierto rango. El valor por defecto es 0.
Dpi <int>	Selecciona la resolución de la imagen de salida, en dots per inch. El valor por defecto es 300.
Word <int>	Selecciona el número total de bytes por error de corrección del codeword. Los valores permitidos son 24, 64, 128, y 255. El valor por defecto es 255.
Ecc <int>	Selecciona el número de bytes de paridad de corrección de errores del codeword. El rango permitido va desde 0 hasta 127 inclusive, y el valor por defecto es 64.
Size <string>	Determina las dimensiones mínimas del bloque de glyph, en términos de filas y columnas de marcas de glyph. El valor por defecto es 16x16.
Grow <string>	Determina como el tamaño y el ecc del bloque de

	glyph deberían cambiar, si fuera necesario, para acomodar los datos codificados. Los valores permitidos son ninguno (none), ancho (width), alto (height), ambos (both), y densidad (density). Ancho y alto permiten al dataglyph crecer a lo largo del eje especificado. Ambos permite al ancho y la altura crecer simultáneamente. La densidad permite al nivel de corrección de errores, decrecer. Si se selecciona ninguno, y los datos no entran en el bloque, con el tamaño y ecc especificado, retorna un error. El valor por defecto es ninguno.
Format <string>	Especifica la forma de la imagen de salida. Los valores permitidos son bmp, font, ps, eps, glyphtonefont, y tif. 'tif' y 'bmp' producen una imagen de mapa de bits. 'ps' produce un postscript utilizando el font standard DG500. 'font' produce un texto ASCII para ser interpretado en el font mencionado. 'eps' produce un estilo simple basado en postscript encapsulado. 'glyphtonefont' produce un texto ASCII para ser interpretado con un font de glyphtone no estándar. El valor por defecto es bmp.
Code <string>	Especifica un esquema logico de codificación. Valores permitidos son *dg500* y pequeño (*small*). dg500 es el formato estandar de Xerox DataGlyph. El formato no estándar small permite codificar bloques de glyph más pequeños que marcas de glyph de 16x16, que es el tamaño mínimo de bloques de dg500. El valor por defecto es dg500.
Image <string>	Especifica una imagen binaria o a escala de grises que modula el grosor de las marcas de glyph. El rango dinámico de la salida de la imagen de glyph será menor que la imagen de entrada, para probar la probabilidad de una decodificación exitosa. Este parámetro acepta formato bmp y tif. Por defecto, no se necesita ninguna modulación.
Tile <string>	Dispersa el bloque de DataGlyph en una superficie de tamaño de MxN marcas de glyph. Por defecto, no se hará esta dispersión.
Out <string>	Especifica un archivo donde colocar la imagen de salida, en lugar de escribirlo en la salida estándar
log	Crea un archivo de log, denominado "dgencode.log", que captura la información acerca del proceso de codificación.

Función *dgdecode*

Modo de uso: dgdecode [opciones] [archivo de entrada]

Opciones:

Help	Muestra un resumen de los comandos.
Id <string>	Licencia requerida para activar el Toolkit de Dataglyph.
Rect <string>	Especifica los límites del rectángulo, como wxh+l+t en coordenadas de pixeles, del bloque de glyph, dentro de la imagen. Si es omitido, se asume que la imagen entera será un solo bloque. Los límites pueden ser aproximados.
Reader <string>	Especifica el front end de la imagen. Opciones

	posbibles son StandardMono y StandardGray. Por defecto, se selecciona automáticamente el lector apropiado.
Cell <float>	Especifica el espacio entre marcas de glyph, en pixeles. Si se omite, el tamaño es automáticamente determinado con la imagen.
Mark <float>	Reemplaza el tamaño de marca esperado.
Angle <float>	Especifica la inclinación del ángulo del bloque de glyph, en grados. Si es omitido, el ángulo es automáticamente determinado desde la imagen.
MaxAngle <float>	Especifica la máxima rotación desde la normal del bloque de glyph, en grados. Los valores válidos van desde 0 a 45, inclusive. El valor por defecto es 22.5
Rotated	Busca una rotación de 0, 0, 90, 70, and 270 grados.
Tiled	Busca específicamente un dataglyph esparcido.
Quiet	No mostrar los resultados de la decodificación.
Shrink	Encoge la imagen por un factor de 2 antes de intentar la decodificación. Esto mejora Esto mejora la confiabilidad si el glyph supera aproximadamente los 15 pixeles.
Data	Lee valores de bloque en lugar de una imagen.
Code <string>	Especifica un esquema lógico de decodificación, dg500 o small. El valor por defecto es dg500.
Log	Crea un archivo de log, denominado "dgdecode.log", que captura la información acerca del proceso de decodificación.
Out <string>	Especifica un archivo en el cual se coloque los datos decodificados, en lugar de escribirlos en una salida estándar.

Ejemplo de uso del *dgencode* y la salida que produce

Texto de entrada para generar un DataGlyph:
ESTO ES UNA PRUEBA MUY GRANDE DE NUMEROS Y LETRAS QUE SERAN ALMACENADOS EN FORMATO DE DATAGLYPHS. HAY ALGUNOS SIGNOS DE PUNTUACION, PERO NO HAY ACENTOS (123456123456123456123456)

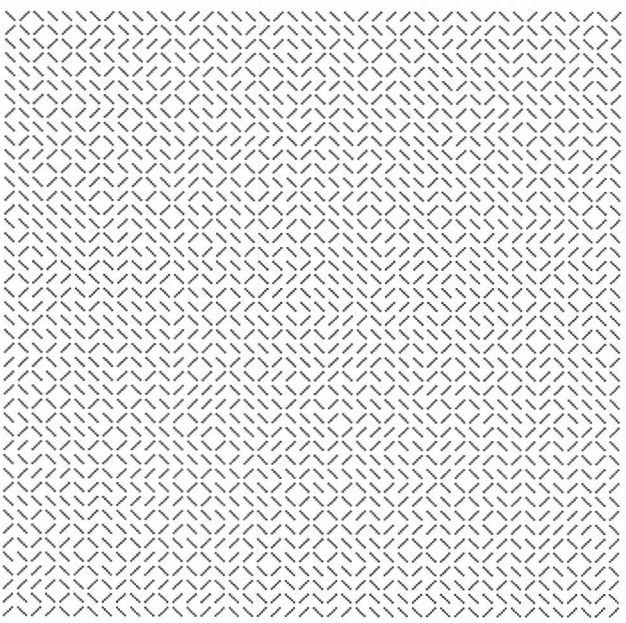


Fig. 3-3. DataGlyph resultante del *dgencode* del ejemplo.

4. Resolución del Problema

La codificación embebida de Dataglyphs en códigos de barras tradicionales consiste en dos grandes etapas, la primera es la de *Codificación*, y la segunda es la de *Decodificación*. En la etapa de *codificación* se recibe como entrada un código de barras tradicional, y un texto a ser codificado en el código embebido. Luego se realiza la fusión de ambos y el resultado queda disponible para ser impreso en el sustrato que se haya elegido. En la etapa de *decodificación* se recibe como entrada un archivo de mapa de bits escaneado y se lo procesa de modo de decodificar la información embebida dentro del símbolo de código de barras.

4.1 Circuito del Problema

En la Fig. 4-1 se muestra el circuito completo del problema a resolver, desde sus parámetros de entrada hasta el resultado.

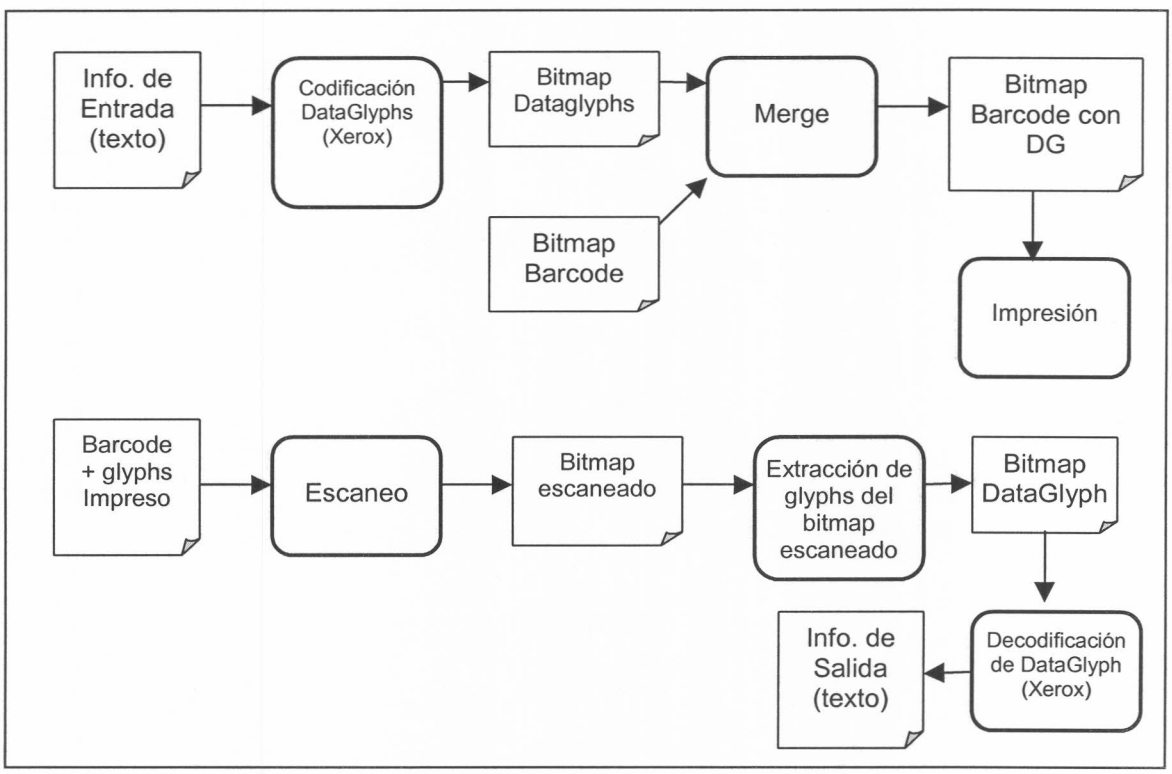


Fig. 4-1. Circuito Completo del Problema a Resolver

Se dividirá el circuito en dos etapas. Se denominará *IDA* a la primera etapa, que comienza con el cuadro "Info. de Entrada (texto)" y que termina en el cuadro "Impresión", y que contiene el proceso de Codificación; se denominará *VUELTA* a la segunda etapa, que comienza con el cuadro "Barcode + glyphs Impreso" y termina con el cuadro "Info. de Salida (texto)", e incluye el proceso de decodificación.

En la etapa de IDA, los parámetros de entrada son dos. El primero de ellos es el texto que se desea codificar embebido, por ejemplo, el ID de un producto, su descripción y su precio. El segundo parámetro es el símbolo de código de barras sobre el cual se codificará el texto anterior. Este parámetro debe respetar el requisito de ser un símbolo de código de barras de una dimensión con una

estructura válida, ya que es importante que tenga suficiente tamaño para su procesamiento. Sin embargo no se estará comprobando explícitamente que así sea. Este requisito es necesario debido a que el algoritmo trabaja suponiendo que lo está haciendo sobre un símbolo de código de barras de una dimensión, y con una longitud de símbolo de uso real. Este símbolo de código de barras debe ser provisto dentro de un archivo de mapa de bits.

A continuación se realiza la codificación del texto de entrada como DataGlyph. Para ello se utilizan las librerías provistas por Xerox que permiten trabajar con todas las características de ésta tecnología. Específicamente, se hace uso de una función denominada *dgencode* a la cual se le ingresan varios parámetros de entrada entre los cuales está el texto a codificar. En particular, se han utilizado los parámetros adicionales de la función en sus valores por defecto, lo cual nos genera como salida un archivo de mapa de bits que contiene lo que técnicamente se denomina un DataGlyph, que es un cuadrado de N x N glyphs que contiene el texto y toda la información de control en forma codificada.

La siguiente etapa es la de *embeber o codificar* el DataGlyph resultante del paso anterior en el símbolo de código de barras. Se utilizará como única característica relevante el valor de cada glyph del DataGlyph, es decir, si representa un cero (0) o un uno (1). Para ello se realizó el proceso *merge* haciendo un recorrido del DataGlyph de izquierda a derecha y de arriba hacia debajo. Por ejemplo, para una porción de 20 glyphs el recorrido será

→ 1 → 2 → 3 → 4 → 5 →
→ 6 → 7 → 8 → 9 → 10 →
→ 11 → 12 → 13 → 14 → 15 →
→ 16 → 17 → 18 → 19 → 20

y uno a uno se van incorporando al símbolo de código de barras. Luego, se genera un archivo de mapa de bits con el resultado esperado.

A modo informativo, para probar el correcto funcionamiento de la lectura del símbolo de código de barras tradicional con el código embebido, se debe realizar una impresión del mismo con tecnología y resolución adecuadas. (Ver Apéndice C)

La etapa de VUELTA comienza con el escaneo de un símbolo de código de barras con glyphs embebidos, cuyo resultado es el parámetro de entrada al proceso de *extracción*.

El proceso de *extracción* es el que contiene la lógica de decodificación. Su primer paso es realizar una cuantización de los valores del archivo de mapa de bits a valores de blanco y negro. Luego se realiza un recorrido de dicho mapa de bits en busca de glyphs, a través de heurísticas.

Este último mapa de bits es el parámetro de entrada para el proceso de decodificación que utiliza la función *dgdecode* de Xerox PARC, y que devolverá el texto originalmente introducido, siguiendo lo enunciado anteriormente, el ID de producto, la descripción y el precio.

4.2 Detalle de los Procesos de IDA

Proceso Merge
Este proceso recibe como parámetro de entrada el archivo de mapa de bits generado con la función *dgencode* de la libreria de Xerox PARC y un archivo de mapa de bits que contiene el símbolo de código de barras de una dimensión sobre el cual se embeberán los glyphs. (Ver Fig. 4-3)

El primer paso del proceso, es detectar el *tamaño de celda óptima* del glyph, dentro del símbolo de código de barras; este proceso sirve para mejorar la capacidad de almacenamiento del símbolo de código de barras.

Luego se dibujan en el símbolo de código de barras uno a uno los glyphs que se encuentran en el archivo de mapa de bits resultado de la función *dgencode*. El recorrido se hace de izquierda a derecha y de arriba hacia abajo, es decir, se lee el primer glyph que se encuentra en el ángulo superior izquierdo y se avanza hacia la derecha hasta finalizar esa fila, luego se continúa desde el primer glyph de la siguiente fila, y así sucesivamente hasta que no queden glyphs por codificar. El dibujado de los glyphs en el mapa de bits del símbolo de código de barras se realiza siguiendo el mismo recorrido. (Ver Fig. 4-4)

Cada glyph es dibujado en el símbolo de código de barras, representando el mismo valor que tenía en la salida del *dgencode*, respetando las siguientes reglas:

- la quiet zone no se utiliza para la inserción de glyphs
- en las barras negras siempre se codifican glyphs
- en las barras blancas se codifican glyphs sólo en las filas impares (1,3,5...)
- se codifica sólo en las barras y espacios cuyo ancho sea mayor o igual al tamaño de celda óptima

Si la cantidad de glyphs del DataGlyph que se desea almacenar no cabe en el mapa de bits del código de barras, no se puede realizar el proceso *merge*.

Características de los DataGlyphs insertados
Si bien el ancho y alto de la celda pueden variar, de acuerdo al tamaño de celda óptima, el grosor no varia, y se mantiene en 2 pixeles.

Si se considera un tamaño de celda N, el alto del glyph será de N-1 y el ancho de N-2. Por ejemplo, para una celda de 5 x 5, el alto del glyph será de 4 y el ancho será de 3. El grosor se mantiene en 2. (Ver figura 4-2)

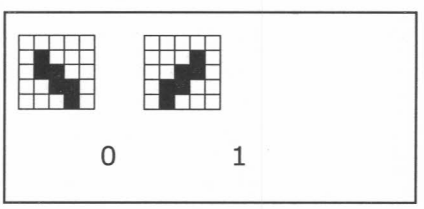


Fig. 4-2. Patrones de glyphs.

Fig. 4-3. Símbolo de código de barras de Code39 sin glyphs.

Fig. 4-4. Símbolo de código de barras de Code39 luego de insertados los glyphs.

Tamaño de celda óptima

El tamaño de celda del glyph dentro del símbolo de código de barras es prefijado dinámicamente de acuerdo a los anchos de las barras del mismo. Una vez prefijado, se mantiene igual para todo el proceso de codificación.

El algoritmo busca un tamaño de celda óptima, que minimice el desperdicio y que permita codificar la mayor cantidad de glyphs posible en el símbolo de código de barras.

Se denomina *desperdicio* a la cantidad de pixeles de un módulo que no son utilizados por ninguna celda de glyph. Esto puede ocurrir por 2 razones:

- el ancho de la celda es mayor que el ancho del módulo, entonces el módulo completo se desperdicia
- el ancho de la barra no es múltiplo del ancho de la celda, entonces el resto de dividir los dos números será la cantidad de pixeles que se desperdiciarán

Para encontrar el tamaño de celda óptima lo primero que se busca es el menor ancho de barra de todas las barras, al que se denomina W. Luego, se almacenan en un vector auxiliar los distintos anchos de barra del símbolo de código de barras, y para cada ancho su cantidad de apariciones.

A continuación, para distintos anchos de celda posibles, se calcula la cantidad de glyphs y el desperdicio para todo el símbolo; el desperdicio que puede resultar necesario en caso de igualdad en la cantidad de información que se puede expresar, para distintos tamaños de celda sugeridos, que van, desde 5 pixeles (sugerido por XEROX como tamaño mínimo de celda), hasta W.

Por último, se elige el tamaño de celda que logra la mejor cantidad de información codificada según el criterio expuesto en el párrafo anterior.

```
// obtener valores iniciales
N = ancho mínimo de barra
A = {distintos anchos de barra del símbolo de códigos de barra}
H = {histograma de A}

// obtener cantidad de información y desperdicio para cada tamaño de celda
para k desde 5 hasta N
        para i en cada valor de A
I_k = I_k + H_i * [N * A_i / k]              // cantidad de información
D_k = D_k + H(A_i) * [(mod(N*A_i,k))]  // cantidad de desperdicio

// obtener tamaño de celda óptima
C = N - 5
```

```
M = 0
para i desde 1 hasta C
        si I_i > I_M
                M = j
        Sino
                si I_i = I_M
                        si D_i < D_M
                                M=j

Celda óptima = I_M
```

Fig. 4-5. Pseudocódigo del algoritmo de Cálculo de Celda Óptima

En la Tabla A-1 del Apéndice A se puede apreciar los resultados obtenidos al aplicar el algoritmo de celda óptima para un mensaje dado, codificado en Code 128.

Para el mensaje "1234 abcd", y para valores de X (ancho de módulo): 5, 6, 7, 8 se obtuvieron los siguientes resultados:

Tabla 4-1. Resultado de Cálculo de Celda Óptima para distintos valores de X.

Valor de X	**5**	**6**	**7**	**8**
Valor Celda Óptima	5	6	5	5
Cantidad de glyphs	123	123	137	165
Desperdicio	0	0	176	159

Barra de control

El último paso de la codificación es la generación de una barra horizontal, que abarca todo el ancho del símbolo de código de barras, incluidas la quiet zones.

La barra tiene 7 pixeles de alto y además contiene una muesca de 5 x 5 ubicada a partir de la tercera fila y de la columna 65*W (ver apéndice E), donde W es el ancho de celda. (Ver Fig. 4-6)

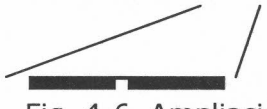


Fig. 4-6. Ampliación parcial de la Muesca en la Barra de Control.

La barra de control garantiza una rotación precisa ya que colabora con el cálculo del ángulo a rotar; también sirve para detectar si la imagen está invertida. Por último, le permite al proceso de *extracción* saber cual es el tamaño de la celda con el que se codificaron los glyphs en el proceso IDA, a través de la muesca que se dibuja en ella.

La barra se coloca en la imagen luego de finalizado el proceso *merge*, y el mapa de bits con el símbolo de código de barras y los glyphs embebidos queda listo para su posterior impresión. (Ver Fig. 4-7) La barra se quita en el circuito de VUELTA cuando se ha enderezado la imagen.

Fig. 4-7. Símbolo de código de barras con glyphs y barra de control.

Cada elemento de la matriz representa la cantidad de pixeles del mismo color (blanco o negro) que aparecen consecutivamente. De esta forma, si en una fila i de la matriz se tienen los valores: 400, 1, 4, 11, 4, 4,..., 402, significa que en la fila i del mapa de bits hay, en forma consecutiva y comenzando por la izquierda, 400 pixeles blancos, 1 negro, 4 blancos, 11 negros, 4 blancos, 4 negros, etc, y finalmente 402 blancos. (Ver recuadro en Fig. 4-12)

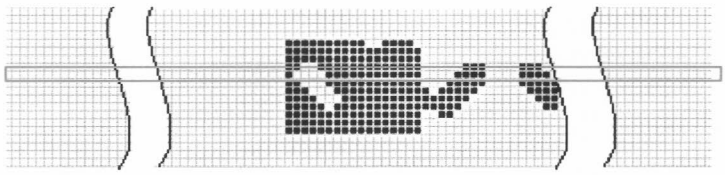


Fig. 4-12. Porción de símbolo de código de barras.

Con la información de la matriz, se puede deducir en que fila y columna de pixeles empieza la primera barra negra, cual es el alto de las barras y en que columna finaliza la ultima barra negra.

Para deducir la ubicación de la primera columna negra del símbolo de código de barras utilizando la matriz, se procede de la siguiente manera:

1. Se obtienen los valores de la matriz para todas las filas
2. Se hace un histograma con los valores de la matriz de la primera columna
3. Se toma el valor que más se repite
4. Ese valor indica en que columna del mapa de bits empieza la primera barra negra
5. Se busca en que fila de la matriz empieza a repetirse ese valor, y donde deja de repetirse. Eso brinda información del alto de la barra

El siguiente dato que se calcula es el alto de un glyph. No debe confundirse con el alto de una celda ya que el alto de un glyph será siempre menor o igual al alto de una celda. Para ello se recorren las 10 primeras *columnas de glyphs* negras. (se utilizará el término *columna de glyphs* para significar una columna cuyo ancho es el tamaño de una celda de glyph) Se utilizan sólo las columnas negras porque éstas siempre tienen glyphs, no como en las columnas blancas donde hay glyphs sólo en las filas impares. Se efectúa un recorrido por cada columna de glyphs, se calcula el tamaño de cada glyph y se cuenta la cantidad de glyphs encontrados. Al final de cada columna se utilizan ambos datos para calcular la altura promedio de un glyph y se redondea para hallar un valor entero. Este mismo procedimiento se repite para cada columna, y al finalizar el recorrido de todo el mapa de bits se halla la altura promedio de un glyph para todo el mapa de bits.

En el ejemplo de la Fig. 4-13, puede observarse que el primer Glyph mide 6 pixeles de alto, el segundo 6, el tercero 7 y el cuarto 6. La altura promedio redondeada es de 6 pixeles.

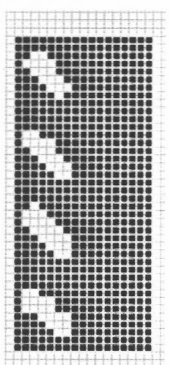


Fig. 4-13. Columna de glyphs.

Por último, se almacena en un vector al que se ha denominado *vector de comienzos* la posición donde comienza cada barra negra o blanca.

Todos estos datos serán de utilidad para recorrer luego el mapa de bits en zonas precisamente delimitadas y con la mayor precisión posible.

Con toda la información obtenida, se está en condiciones de comenzar el recorrido final. Éste se realiza por columnas de glyphs, y dentro de la columna de glyphs, por celdas. Luego de finalizada cada columna se pasa a la columna de glyphs inmediata siguiente.

Hay que tener en cuenta que:

- Los glyphs dentro de una misma columna son equidistantes
- Los glyphs están alineados desde la primera columna a la ultima
- Una barra puede tener más de una columna. De hecho, una barra que contenga sólo una columna se denomina barra angosta
- No todos los tamaños de las barras son múltiplos entre sí, una barra podría tener 5 pixeles y otra 7

Estas últimas características son tomadas en cuenta a la hora de recorrer el símbolo de código de barras.

Para pasar a la próxima columna, se utiliza la información del vector de comienzos. El problema surge cuando se tiene desperdicio y hay que decidir si ese desperdicio, en realidad, no es tal, sino, otro módulo. (Ver Fig. 4-14)

Las celdas están delimitadas en los cuadros, de 8x8. Lo que está fuera de los marcos es el desperdicio.

Desperdicio

Fig. 4-14. Identificación de información y desperdicio.

La heurística que se usa para determinar si se está ante una columna de glyhps y no una de desperdicio es la siguiente:

- el ancho de la columna en conflicto debe superar el 75% del ancho de la celda
- debe haber glyphs en la columna

Cada glyph detectado se guarda en una matriz como uno, cero o espacio, según corresponda, respetando la ubicación del glyph dentro del símbolo de código de barras, es decir, la primera columna de glyph corresponde a la primera columna de la matriz, y así sucesivamente. El hecho de almacenar un espacio en la matriz indica que en esa celda no hay ningún glyph, como sucede en el caso de las barras blancas filas pares.

Como ejemplo, se utiliza la siguiente porción del mapa de bits con glyphs. (Ver Fig. 4-15)

Fig. 4-15. Porción de mapa de bits con glyphs.
La matriz que se genera se muestra en la Tabla 4-2.

Tabla 4-2. Matriz con los elementos decodificados (0, 1 o espacio).

0	1	0	0	0	0	0	1	0	0	0
0							1	0		
0	0	1	1	0	0	0	1	1	0	0
0							1	1		
1	0	1	0	0	1	0	0	1	1	1
1							0	1		
0	0	0	0	1	0	0	0	0	1	1
0							1	0		
0	0	0	0	0	1	1	1	0	1	0

Una vez recorrido el mapa de bits y detectado los glyphs se contabiliza la cantidad encontrada. La misma debe ser igual a un cuadrado perfecto, ya que en el proceso *Codificación Dataglyphs (Xerox)* de la etapa IDA se genera siempre una matriz cuadrada de N x N glyphs. Si el número resultante no es un cuadrado perfecto se realiza un proceso de búsqueda mediante heurísticas, para completar o eliminar información según corresponda.

Acerca de la detección de glyphs
Para la detección de glyphs se utiliza la siguiente técnica. Durante el recorrido de las columnas de glyphs, en cada celda se trabaja con una ventana de glyph cuyo ancho y alto es del alto calculado de un glyph. A su vez, la ventana de glyph se divide en cuatro partes que de denominarán *subventanas* como se muestra en la Fig. 4-16. Como el tamaño de la ventana es menor que el de la celda, se va deslizando aquella por toda la celda, de modo de determinar si existe un glyph en esa celda, y en caso afirmativo, si ese glyph corresponde a un cero o a un uno.

1	2
3	4

Fig. 4-16. Modelo de ventana de glyph.

En la Fig. 4-17 se observa el recuadro exterior que corresponde a una celda, y el recuadro interior que corresponde a una ventana de glyph, dividido en 4 partes iguales.

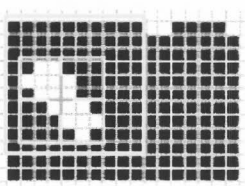


Fig. 4-17. Celda y ventana de glyph.

La heurística más importante es la que determina si se está en presencia o no de un glyph, y consiste en encontrar una posición de ventana donde el conjunto de pixeles que se hallan en ella determinen un importante parecido entre las *subventanas* 1 y 4, o entre las subventanas 2 y 3.

En el caso de la figura, se observa claramente que la subventana 1 tiene 7 pixeles blancos y la subventana 4 tiene 6 pixeles blancos. En consecuencia, se está en

presencia de un glyph. Ahora resta saber si es un uno o un cero. Luego, la subventana 2 tiene 2 pixeles blancos y la subventana 3 tiene sólo 1. Por lo tanto, el par de subventanas 1 y 4 tienen ventaja sobre el par de subventanas 2 y 3, por lo que se deduce que el glyph detectado es un cero (0).

De esta manera se recorre todo el mapa de bits almacenando en una matriz los valores de glyphs hallados.

Decodificación de la matriz cuadrada

A partir de la matriz mencionada se genera una matriz cuadrada de N x N similar a la Fig. C-9, que se almacenará como mapa de bits. Este archivo será parámetro de entrada del proceso *dgdecode* y se obtendrá como resultado el texto ingresado al comienzo de todo el circuito.

Restricciones de la aplicación

Este trabajo presenta restricciones, algunas de las cuales no se han resuelto, y otras se han resuelto utilizando soluciones de compromiso, para no desviar la atención respecto del problema principal a resolver. Estas restricciones se describen en el apéndice E.

5. Medidas de la Información

De modo de exponer la contribución que se realiza en este trabajo, se han seleccionado tres indicadores que expresan la mejora que se ha producido debido a la incorporación de la tecnología de glyphs a los símbolos de código de barras.

Expresividad
Este indicador muestra la mejora en la cantidad de información que puede ser expresada en un símbolo de código de barras con glyphs embebidos, con relación a un símbolo de código de barras sin glyphs.

Como se vio anteriormente, un mensaje codificado con cualquier simbología ocupa una cierta cantidad de módulos a lo ancho y a lo alto. Sin embargo, la altura de las barras y espacios no lleva información. Un mensaje corto de un caracter se puede codificar, por ejemplo en Code 39 con una relación ancho/angosto de 2:1, como un símbolo que ocupa 38 módulos de ancho más las quiet zones. Si se codifica a una relación de un glyph por módulo (el peor caso de codificación permitido) se puede codificar 38 glyphs es decir 38 bits, por lo tanto 4,5 caracteres ASCII, mucho más que los 5,43 bits que se requieren para codificar el único caracter en Code 39.

Pero el valor de 38 bits representa sólo la expresividad en una sola fila de glyphs. El alto mínimo del símbolo debe ser de un 15% del ancho, por lo que debe ser de 6 módulos. Luego, tomando un caracter poco conveniente del juego de caracteres de Code 39 como el '$', y haciendo los cálculos correspondientes, se pueden expresar 3 filas de 38 glyphs y 3 filas de 19 glyphs, lo que da un total de 171 glyphs o 171 bits, lo que hace un total de más de 21 caracteres ASCII. En este ejemplo, la expresividad mínima es de más de 21 veces la expresividad del símbolo de código de barras. Para un mensaje largo en la misma simbología, por ejemplo de 20 caracteres, la expresividad mínima de un símbolo de código de barras con glylphs es de 8664 bits, es decir, 1083 caracteres ASCII. Para un símbolo de código de barras de un producto de supermercado, codificado por ejemplo en UPC, la expresividad del símbolo es de 12 caracteres mientras que la con glyphs la expresividad mínima es de más de 2625 bits o más de 328 caracteres ASCII.

Si se realizan cálculos para todas las simbologías de una dimensión se logran los mismos resultados contundentes.

Eficiencia
Este indicador muestra la expresividad medida con relación al espacio ocupado por el símbolo en el sustrato, ya sea en forma lineal o en superficie.

La eficiencia lineal es una medida típicamente utilizada para comparar distintas simbologías de una dimensión.

Como se vio en el capítulo 2.2, las eficiencias para las distintas simbologías de una dimensión van desde más de un 25% hasta casi un 36%.

Los glyphs embebidos tienen dos elementos en el juego de caracteres, 0 y 1. Por lo que el cálculo de Información del capítulo 2.2, o lo que es lo mismo la cantidad de bits de datos es igual a 1. Además, en el peor caso, la longitud en módulos de cada elemento del juego de datos es de 1 módulo. De esta manera, como los glyphs no tienen caracteres de control, la eficiencia lineal bruta es igual a la eficiencia lineal neta. Finalmente, la eficiencia lineal neta de los glyphs es del 100%.

El otro indicador es el de eficiencia en área que se utiliza normalmente para comparar simbologías de códigos de barras de dos dimensiones, o bien para mostrar la diferencia entre la eficiencia de una simbología de dos dimensiones contra la de una simbología una dimensión, o en este caso, la de un símbolo de código de barras con glyphs contra la de la misma simbología sin glyphs.

El indicador de eficiencia en área puede verse en el capítulo 2.2, y resulta de la división entre la cantidad de bits de datos y la cantidad de módulos que ocupan en superficie.

Utilizando los ejemplos anteriores de la medición de expresividad, un símbolo de Code 39 que codifica un mensaje de un caracter tiene una eficiencia en área de 2,4%. Luego, un símbolo de Code 39 que codifica un mensaje de 20 caracteres tiene una eficiencia en área de 0,9%. En el caso de un símbolo UPC de supermercado, la eficiencia en área es del orden de 0,1%.

Si se codifican glyphs sobre el símbolo de UPC, la eficiencia en área de los glyphs seleccionando el peor caso posible, es como mínimo de 65,8% pudiendo llegar a eficiencias superiores al 110%.

Densidad
Una medida asociada a la eficiencia en área es la densidad, que muestra también la manera en la que se aprovecha el espacio pero en este caso con respecto al total de pixeles del símbolo.

Para calcular densidad, se utiliza la fórmula:

$$densidad = \frac{bitsDeDatos}{pixeles}$$ [Pav92]

Para los mismos ejemplos anteriores, las densidades de los glyphs dentro de los símbolos de código de barras respectivos son las siguientes:
- Code 39, mensaje de 1 caracter: 0,03 bits / pixel
- Code 39, mensaje de 20 caracteres: 0,0283 bits / pixel
- UPC, mensaje de 12 caracteres: 0,0263 bits / pixel

Como medida comparativa, se muestran las densidades de otras simbologías, en particular todas de dos dimensiones:
- Code 49: 0,0527 bits / pixel
- Code 16K 0,0567 bits / pixel
- Identcode MLC-2D: 0,0148 bits / pixel
- PDF417: 0,1364 bits / pixel
- Softstrip: 0,500 bits / pixel
- Vericode: 0,529 bits / pixel
- Datacode: 0,549 bits / pixel

La mayoría de estas simbologías no han sido nombradas en el presente trabajo, sin embargo son de uso habitual.

No es objetivo del presente trabajo comparar los resultados obtenidos con glyphs embebidos en símbolos de códigos de barras en una dimensión con simbologías de dos dimensiones, sin embargo resulta ilustrativo apreciar las diferencias entre unos y otros.

En el Apéndice D pueden verse algunos ejemplos de mensajes en Code 39 y Code 128, y sus distintas medidas de información.

6. Conclusiones

El presente trabajo muestra que es posible incrementar la capacidad expresiva de los símbolos de código de barras tradicionales de una dimensión a través de su integración con la tecnología denominada DataGlyphs desarrollada por Xerox PARC.

En la mayoría de las implementaciones, o bien en las más difundidas como la de los supermercados, toda la información de un producto como por ejemplo la marca, el tipo, la descripción del mismo, el gramaje o volumen y el precio, está almacenada en una base de datos a la que se accede utilizando como clave el código de identificación de producto que expresa el símbolo de código de barras impreso en su envase. [Pav92]

Con la integración de la tecnología de DataGlyphs a través de la inserción de pequeños elementos denominados glyphs en el símbolo, el poder expresivo se incrementa notablemente. Toda la información antes mencionada puede incluirse en el mismo símbolo de código de barras sin afectar la lectura de éste por los dispositivos lectores de códigos de barras usuales, permitiendo la decodificación de los glyphs por parte de dispositivos de lectura especiales o por combinaciones de hardware y software como la desarrollada en el presente trabajo.

Los símbolos de códigos de barras tradicionales consisten en barras y espacios paralelos que almacenan información en sus tamaños de ancho. Su altura no aporta información, mas bien representa un mecanismo de redundancia utilizado para mejorar la efectividad de la lectura del símbolo en ángulos más amplios. [Pav92] A su vez, estos símbolos son impresos a una cierta resolución de impresión y son escaneados a otra resolución de lectura.

La impresión de los símbolos de código de barras puede ser una fuente de interferencias para los dispositivos lectores ya que pueden crearse manchas blancas en las barras negras o manchas negras en los espacios blancos. A su vez, los lectores detectan la información a cierta resolución de lectura que posee el dispositivo. Las especificaciones formales indican cual es la tolerancia que se puede tener en la impresión, teniendo en cuenta que los dispositivos de lectura no deben llegar a detectar las imperfecciones de impresión. Si se imprime el símbolo de código de barras dentro de los parámetros indicados en la especificación de la simbología no habrá problemas en su decodificación. [Pal91]

Otra fuente de imperfecciones en el símbolo de código de barras es el maltrato que pueda sufrir el sustrato donde está impreso. En este caso, no existe una mitigación específica del problema, sólo se debe intentar mantener el sustrato en las mejores condiciones posibles. [Pal91]

La tecnología denominada DataGlyph consiste en la codificación de elementos visualmente no obstructivos que pueden incorporarse en distintos elementos subyacentes y no ser detectados visualmente. En algunos casos pueden ellos mismos conformar imágenes a través de su manipulación en distintos tamaños en altura o grosor e incluso en color. Una gran ventaja es que uno de estos elementos, denominados glyphs, puede dibujarse en tan sólo 3 pixeles lo cual lo hace ideal para ser introducidos en las barras o espacios de los símbolos de código de barras. [WWW1]

Asimismo, la tecnología DataGlyphs es muy robusta desde el punto de vista de la decodificación, de manera que su utilización le asegura al usuario una gran probabilidad de éxito incluso ante importantes deterioros en el conjunto de glyphs. [WWW1]

El mercado minorista y otras diversas industrias, utilizan el símbolo de código de barras de una dimensión como un estándar desde hace más de 10 años. Cada compañía que participa en estos mercados o industrias se ha acoplado sin alternativas al uso de ésta tecnología adquiriendo productos comerciales para la impresión y para la lectura de dichos códigos de barras. Muchas de ellas verían con agrado la incorporación de tecnologías de códigos de barras de mayor poder expresivo para distintas aplicaciones posibles. Sin embargo, aunque han aparecido nuevas simbologías de códigos de barras de dos dimensiones con importantes incrementos expresivos, ningún mercado migró a alguna de ellas debido a que sería necesaria una importante inversión en infraestructura tecnológica.

El resultado de este trabajo, es decir, la integración entre la tecnología de símbolos de código de barras de una dimensión y los glyphs extraídos de la tecnología de DataGlyphs, aparece como un camino natural de evolución hacia simbologías de mayor poder expresivo en las compañías. A través de una gradual adquisición de infraestructura se puede adoptar en forma igualmente gradual la tecnología presentada en este trabajo, a la vez que se puede seguir utilizando en paralelo la tecnología de códigos de barras tradicional.

7. Trabajos Futuros

Un trabajo previo a la realización de cualquier avance sobre este tema debe ser el de eliminar las restricciones señaladas en el apéndice E de este trabajo.

Si bien existen productos comerciales de hardware para la lectura de DataGlyphs, los mismos lo hacen bajo la habitual disposición reticular de los glyphs. Un trabajo futuro sería diseñar un dispositivo de hardware que permita la lectura de glyphs desde un símbolo de código de barras, donde la distribución de los glyphs es la que se expuso en el presente trabajo. Asimismo, el mismo dispositivo debería ser capaz de detectar los códigos de barras tradicionales.

Apéndice A

Ejemplo del algoritmo de Cálculo de Celda Óptima.

En este apéndice se mostrará como se realiza el cálculo de celda óptima mediante un ejemplo con el mensaje "1234 abcd" utilizando para ello la simbología Code 128.

La codificación completa del mensaje mencionado debido al agregado de los caracteres de control es la siguiente:

"'StartC', '12', '34','CodeB', '<space>', 'a', 'b', 'c', 'd', '<check-digit>', 'STOP'"

La simbología Code 128 consta de 3 juegos de caracteres denominados A, B, y C. Algunos de ellos codifican mejor algunas combinaciones de caracteres que otros, y por otro lado algunos caracteres están representados sólo en cierto juego de caracteres. Code 128 tiene la particularidad de tener ciertos caracteres que permiten cambiar de juego en el medio de la codificación del mensaje; esto es lo que se ha hecho en este ejemplo.

El caracter 'StartC' indica el comienzo del símbolo y también indica que los siguientes caracteres serán del juego C. Luego se codifican los caracteres '12' y '34'. A continuación, el caracter 'CodeB' indica que los caracteres subsiguientes pertenecen al juego B. Luego se codifican los caracteres restantes del mensaje. Véase que todos los caracteres tienen una longitud de 11 módulos menos el caracter 'STOP' que tiene una longitud especial de 13 módulos. (Ver Apéndice B)

La primera columna de la Tabla A-2, denominada "Módulos", muestra la cantidad de módulos que ocupa cada barra y espacio de cada caracter del mensaje. Por ejemplo, las primeras 6 filas <2, 1, 1, 2, 3, 2> identifican al caracter 'StartC', las siguientes 6 <1, 1, 2, 2, 3, 2> identifican al caracter '12', y así sucesivamente. Cada ejemplo de la tupla indica <barra, espacio, barra, espacio, barra, espacio>.

Los valores de X señalados son los que indican los distintos anchos de módulo que se utilizaron para el ejemplo. El contenido de las casillas debajo de cada columna de X = n surge de multiplicar X por la cantidad de módulos para cada barra o espacio de cada caracter del mensaje. Tomando las primeras filas, se obtiene:

2x5=10,	2x6=12,	2x7=14,	2x8=16
1x5=5,	1x6=6,	1x7=7,	1x8=8
1x5=5,	1x6=6,	1x7=7,	1x8=8

Estos valores indican el espacio en pixeles que ocupa cada barra o espacio de cada caracter del mensaje.

Luego, para cada valor de X y para cada tamaño de celda posible (ver "Tamaño de Celda vs. Valor de X" en la Tabla A-2), se muestra cuantos glyphs o bits de información entran para esa fila y en esa barra o espacio - se indica como (i)-, y cuanto desperdicio en pixeles se tiene para esa fila y en esa barra o espacio - se indica como (d)-, utilizando los mismos principios del algoritmo de celda óptima. (Fig. 4-14)

La última fila es la de totales. En las columnas 2 a 5 se puede ver el total de pixeles que ocupa el mensaje para cada valor de ancho de módulo X, y a continuación el total de glyphs que permite almacenar el símbolo en una fila (i), con el desperdicio respectivo (d), para un X y un tamaño de celda dados.

Las columnas de color gris indican las que tienen celda óptima, por ejemplo para un tamaño de ancho de módulo de 8 pixeles, una celda de 5 pixeles es la que puede almacenar mayor cantidad de información (bits o glyphs), a pesar del desperdicio que tiene. Se presenta un resumen de los resultados finales en la tabla A-1.

Tabla A-1. Resultados de tamaño de celda óptima para cada valor de X

	Valor de X (ancho de módulo)			
	5	***6***	***7***	***8***
Tamaño de Celda Óptima	5	6	5	5

Tabla A-2. Cálculo de Celda Óptima en Code 128, Mensaje: "1234 abcd"
(Ref: i = Nro. de glyphs, d = desperdicio, C = tamaño de celda, X = ancho de módulo)

Módulos	*Valor de X*				*Tamaño de Celda vs. Valor de X*																			
					C = 5								*C = 6*						*C = 7*				*C = 8*	
					X = 5		*X = 6*		*X = 7*		*X = 8*		*X = 6*		*X = 7*		*X = 8*		*X = 7*		*X = 8*		*X = 8*	
	5	6	7	8	i	d	i	d	i	d	i	d	i	d	i	d	i	d	i	d	i	d	i	d
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
4	20	24	28	32	4	0	4	4	5	3	6	2	4	0	4	4	5	2	4	0	4	4	4	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
4	20	24	28	32	4	0	4	4	5	3	6	2	4	0	4	4	5	2	4	0	4	4	4	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
4	20	24	28	32	4	0	4	4	5	3	6	2	4	0	4	4	5	2	4	0	4	4	4	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
4	20	24	28	32	4	0	4	4	5	3	6	2	4	0	4	4	5	2	4	0	4	4	4	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
4	20	24	28	32	4	0	4	4	5	3	6	2	4	0	4	4	5	2	4	0	4	4	4	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
3	15	18	21	24	3	0	3	3	4	1	4	4	3	0	3	3	4	0	3	0	3	3	3	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
1	5	6	7	8	1	0	1	1	1	2	1	3	1	0	1	1	1	2	1	0	1	1	1	0
2	10	12	14	16	2	0	2	2	2	4	3	1	2	0	2	2	2	4	2	0	2	2	2	0
Tot	620	744	868	992	**123**	0	**123**	123	**137**	176	**165**	159	**123**	0	**123**	123	**137**	162	**123**	0	**123**	123	**123**	0

Apéndice B

Tablas de Símbolos de Code 128 y Code 39

Tabla B-1. Code 128 – Juego de Caracteres A, B y C.

Valor	Código tipo A	Código tipo B	Código tipo C	Patrón Barras/ Espacios B S B S B S	Valor	Código tipo A	Código tipo B	Código tipo C	Patrón Barras/ Espacios B S B S B S
0	SP	SP	0	2 1 2 2 2 2	54	V	V	54	3 1 1 1 2 3
1	!	!	1	2 2 2 1 2 2	55	W	W	55	3 1 1 3 2 1
2	"	"	2	2 2 2 2 2 1	56	X	X	56	3 3 1 1 2 1
3	#	#	3	1 2 1 2 2 3	57	Y	Y	57	3 1 2 1 1 3
4	$	$	4	1 2 1 3 2 2	58	Z	Z	58	3 1 2 3 1 1
5	%	%	5	1 3 1 2 2 2	59	[	[	59	3 3 2 1 1 1
6	&	&	6	1 2 2 2 1 3	60	\	\	60	3 1 4 1 1 1
7	'	'	7	1 2 2 3 1 2	61	]	]	61	2 2 1 4 1 1
8	(	(	8	1 3 2 2 1 2	62	^	^	62	4 3 1 1 1 1
9	)	)	9	2 2 1 2 1 3	63	_	_	63	1 1 1 2 2 4
10	*	*	10	2 2 1 3 1 2	64	NUL	`	64	1 1 1 4 2 2
11	+	+	11	2 3 1 2 1 2	65	SOH	a	65	1 2 1 1 2 4
12	,	,	12	1 1 2 2 3 2	66	STX	b	66	1 2 1 4 2 1
13	-	-	13	1 2 2 1 3 2	67	ETX	c	67	1 4 1 1 2 2
14	.	.	14	1 2 2 2 3 1	68	EOT	d	68	1 4 1 2 2 1
15	/	/	15	1 1 3 2 2 2	69	ENQ	e	69	1 1 2 2 1 4
16	0	0	16	1 2 3 1 2 2	70	ACK	f	70	1 1 2 4 1 2
17	1	1	17	1 2 3 2 2 1	71	BEL	g	71	1 2 2 1 1 4
18	2	2	18	2 2 3 2 1 1	72	BS	h	72	1 2 2 4 1 1
19	3	3	19	2 2 1 1 3 2	73	HT	i	73	1 4 2 1 1 2
20	4	4	20	2 2 1 2 3 1	74	LF	j	74	1 4 2 2 1 1
21	5	5	21	2 1 3 2 1 2	75	VT	k	75	2 4 1 2 1 1
22	6	6	22	2 2 3 1 1 2	76	FF	I	76	2 2 1 1 1 4
23	7	7	23	3 1 2 1 3 1	77	CR	m	77	4 1 3 1 1 1
24	8	8	24	3 1 1 2 2 2	78	SO	n	78	2 4 1 1 1 2
25	9	9	25	3 2 1 1 2 2	79	SI	o	79	1 3 4 1 1 1
26	:	:	26	3 2 1 2 2 1	80	DLE	p	80	1 1 1 2 4 2
27	;	;	27	3 1 2 2 1 2	81	DC1	q	81	1 2 1 1 4 2
28	<	<	28	3 2 2 1 1 2	82	DC2	r	82	1 2 1 2 4 1
29	=	=	29	3 2 2 2 1 1	83	DC3	s	83	1 1 4 2 1 2
30	>	>	30	2 1 2 1 2 3	84	DC4	t	84	1 2 4 1 1 2
31	?	?	31	2 1 2 3 2 1	85	NAK	u	85	1 2 4 2 1 1
32	@	@	32	2 3 2 1 2 1	86	SYN	v	86	4 1 1 2 1 2
33	A	A	33	1 1 1 3 2 3	87	ETB	w	87	4 2 1 1 1 2
34	B	B	34	1 3 1 1 2 3	88	CAN	x	88	4 2 1 2 1 1
35	C	C	35	1 3 1 3 2 1	89	EM	y	89	2 1 2 1 4 1
36	D	D	36	1 1 2 3 1 3	90	SUB	z	90	2 1 4 1 2 1
37	E	E	37	1 3 2 1 1 3	91	ESC	{	91	4 1 2 1 2 1
38	F	F	38	1 3 2 3 1 1	92	FS	|	92	1 1 1 1 4 3
39	G	G	39	2 1 1 3 1 3	93	GS	}	93	1 1 1 3 4 1
40	H	H	40	2 3 1 1 1 3	94	RS	~	94	1 3 1 1 4 1
41	I	I	41	2 3 1 3 1 1	95	US	DEL	95	1 1 4 1 1 3
42	J	J	42	1 1 2 1 3 3	96	FNC 3	FNC 3	96	1 1 4 3 1 1
43	K	K	43	1 1 2 3 3 1	97	FNC 2	FNC 2	97	4 1 1 1 1 3
44	L	L	44	1 3 2 1 3 1	98	SHIFT	SHIFT	98	4 1 1 3 1 1
45	M	M	45	1 1 3 1 2 3	99	CODE C	CODE C	99	1 1 3 1 4 1
46	N	N	46	1 1 3 3 2 1	100	CODE B	FNC 4	CODE B	1 1 4 1 3 1
47	O	O	47	1 3 3 1 2 1	101	FNC 4	CODE A	CODE A	3 1 1 1 4 1
48	P	P	48	3 1 3 1 2 1	102	FNC 1	FNC 1	FNC 1	4 1 1 1 3 1
49	Q	Q	49	2 1 1 3 3 1	103	Start A	Start A	Start A	2 1 1 4 1 2
50	R	R	50	2 3 1 1 3 1	104	Start B	Start B	Start B	2 1 1 2 1 4
51	S	S	51	2 1 3 1 1 3	105	Start C	Start C	Start C	2 1 1 2 3 2
52	T	T	52	2 1 3 3 1 1	106	Stop	Stop	Stop	2 3 3 1 1 1 2
53	U	U	53	2 1 3 1 3 1					

Tabla B-2. Code 39.

Caracter	Patrón Barras/ Espacios	Caracter	Patrón Barras/ Espacios
	B S B S B S B S B		B S B S B S B S B
0	1 1 1 3 3 1 3 1 1	M	3 1 3 1 1 1 1 3 1
1	3 1 1 3 1 1 1 1 3	N	1 1 1 1 3 1 1 3 3
2	1 1 3 3 1 1 1 1 3	O	3 1 1 1 3 1 1 3 1
3	3 1 3 3 1 1 1 1 1	P	1 1 3 1 3 1 1 3 1
4	1 1 1 3 3 1 1 1 3	Q	1 1 1 1 1 1 3 3 3
5	3 1 1 3 3 1 1 1 1	R	3 1 1 1 1 1 1 3 3
6	1 1 3 3 3 1 1 1 1	S	1 1 3 1 1 1 3 3 1
7	1 1 1 3 1 1 3 1 3	T	1 1 1 1 3 1 3 3 1
8	3 1 1 3 1 1 3 1 1	U	3 3 1 1 1 1 1 1 3
9	1 1 3 3 1 1 3 1 1	V	1 3 3 1 1 1 1 1 3
A	3 1 1 1 1 3 1 1 3	W	3 3 3 1 1 1 1 1 1
B	1 1 3 1 1 3 1 1 3	X	1 3 1 1 3 1 1 1 3
C	3 1 3 1 1 3 1 1 1	Y	3 3 1 1 3 1 1 1 1
D	1 1 1 1 3 3 1 1 3	Z	1 3 3 1 3 1 1 1 1
E	3 1 1 1 3 3 1 1 1	-	1 3 1 1 1 1 3 1 3
F	1 1 3 1 3 3 1 1 1	.	3 3 1 1 1 1 3 1 1
G	1 1 1 1 1 3 3 1 3	;	1 3 3 1 1 1 3 1 1
H	3 1 1 1 1 3 3 1 1	*	1 3 1 1 3 1 3 1 1
I	1 1 3 1 1 3 3 1 1	$	1 3 1 3 1 3 1 1 1
J	1 1 1 1 3 3 3 1 1	/	1 3 1 3 1 1 1 3 1
K	3 1 1 1 1 1 1 3 3	+	1 3 1 1 1 3 1 3 1
L	1 1 3 1 1 1 1 3 3	%	1 1 1 3 1 3 1 3 1

Apéndice C

Pruebas de lectura de Código de Barras

Valores fijos:

- Simbología utilizada: Code 39 a una relación ancho/angosto de 3:1
- Texto del símbolo de código de barras: TESIS03
- Texto para glyphs: Propuesta de Tesis: Codificación de información embedded en código de barras convencionales. Autores: Eduardo Raichman (LU 231/89), Hernán Podolsky (LU 659/88). Director: Eduardo Rodríguez
- Cantidad de caracteres del texto para glyphs: 188
- Cantidad de glyphs resultante de la función *dgencode*: 2304 (48 x 48)
- Dispositivo de Lectura de Código de Barras: SocketScan de Socket Communications (www.socketcom.com). Es un lector de tipo In-hand card, con una tarjeta PCMCIA
- Dispositivo de Impresión: Impresora Hewlett-Packard LaserJet 8000N
- Cantidad de muestras tomadas: entre 10 y 15

Valores variables:

- *Grosor del glyph*
- *Altura del glyph*
- *Densidad*
- *Barras blancas completas/incompletas*
- *Resolución de Impresión*

A continuación se explicará cada uno de los valores variables.

Grosor del glyph

Se utilizaron 2 tipos de grosor: de 1 pixel y de 2 pixeles. Observar como se ven en la Fig. C-1:

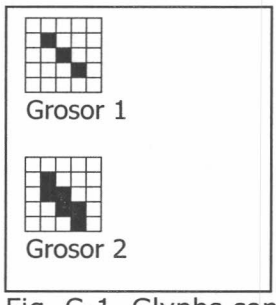


Fig. C-1. Glyphs con grosor 1 y 2.

Altura del glyph
Se utilizaron 2 tamaños para la altura: 3 pixeles y 4 pixeles. Observar como se ven en la Fig. C-2:

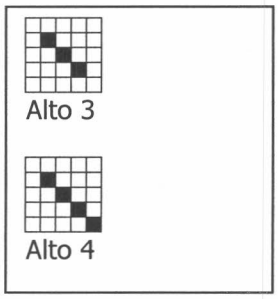


Fig. C-2. Glyphs con altura 3 y 4.

Densidad
La densidad se expresa como la cantidad de módulos que deben separar como mínimo a los glyphs en una misma fila. Se utilizaron 2 densidades: cero y uno. Cero significa que no hay separación y uno que habrá 1 módulo de separación entre ellos.

A continuación se presentan ejemplos de esto último.
En la Fig. C-3 se ve un extracto de un símbolo de código de barras, con densidad 0, es decir, todos los glyphs están uno al lado del otro. Como característica adicional, se observa un alto igual a 3 y un grosor de 1 pixel.

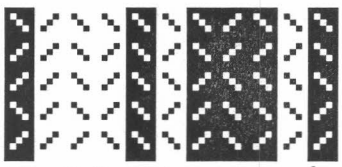


Fig. C-3. Extracto de símbolo de código de barras con densidad 0.

En la Fig. C-4 se observa que cada glyph en una misma fila está separado por un módulo. El extracto tiene 11 módulos con los cuales se conforman 4 barras negras y 3 espacios en blanco o barras blancas. Se toma como ejemplo la primera fila. Comienza con un glyph en la primera barra. La segunda barra tiene un glyph en el medio, ya que un módulo debe separar al glyph anterior. El mismo razonamiento se aplica al resto de la fila.

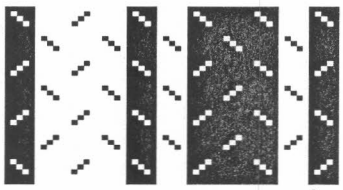


Fig. C-4. Extracto de símbolo de código de barras con densidad 1.

Barras Blancas Completas /Incompletas
Otra variante que se utilizó fue no colocar glyphs en las barras blancas en las filas pares. A esto se lo denomina barra blanca incompleta. Como ejemplo de incompleta se presenta la Fig. C-5, donde se observa en las filas pares que en las barras blancas no hay glyphs.

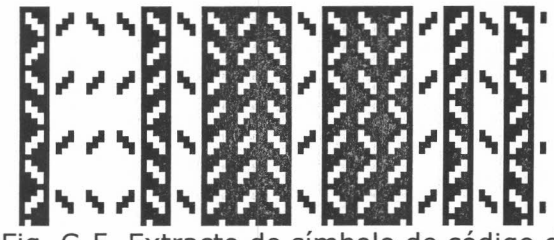


Fig. C-5. Extracto de símbolo de código de barras con Blancos Incompletos.

Resolución de Impresión
Se utilizaron 2 resoluciones que se miden en dots per inch (dpi): 300 y 600.

La Tabla C-1 muestra los resultados obtenidos durante las pruebas de lectura de símbolos de códigos de barras utilizando las combinaciones razonables de los valores fijos y variables recién explicados.

Tabla C-1. Resultados de las pruebas de lectura.

Caso	Blancos completo	Grosor glyph	Altura glyph	Densidad	Resolución Impresión	Lectura
1	SI	2	4	0	300	NOT OK
2	SI	2	4	0	600	NOT OK
3	SI	2	3	0	300	NOT OK
4	SI	2	3	0	600	OK
5	SI	1	4	0	300	OK
6	SI	1	4	0	600	OK
7	SI	1	3	0	300	OK
8	SI	1	3	0	600	OK
9	NO	2	4	0	300	NOT OK
10	NO	2	4	0	600	OK
11	NO	2	3	0	300	REGULAR
12	NO	2	3	0	600	OK
13	NO	1	4	0	300	OK
14	NO	1	4	0	600	OK
15	NO	1	3	0	300	REGULAR
16	NO	1	3	0	600	OK
17	SI	2	4	1	300	REGULAR
18	SI	2	4	1	600	OK
19	SI	2	3	1	300	REGULAR
20	SI	2	3	1	600	OK
21	SI	1	4	1	300	OK
22	SI	1	4	1	600	OK
23	SI	1	3	1	300	OK
24	SI	1	3	1	600	OK
25	NO	2	4	1	300	REGULAR
26	NO	2	4	1	600	OK
27	NO	2	3	1	300	OK
28	NO	2	3	1	600	OK
29	NO	1	4	1	300	OK
30	NO	1	4	1	600	OK
31	NO	1	3	1	300	OK
32	NO	1	3	1	600	OK

Análisis de los resultados obtenidos en las pruebas de lectura
Como se observa en la Tabla C-1, se han obtenido 3 tipos de resultados.

- NOT OK: todas las lecturas resultaron insatisfactorias
- OK: todas las lecturas resultaron satisfactorias
- REGULAR: algunas lecturas resultaron insatisfactorias

El primer y el tercer caso han resultado insatisfactorios debido a varios factores: el grosor y la altura del glyph, la densidad y el hecho que se imprimen glyphs en forma completa en las barras blancas. Respecto a la resolución de impresión, es importante destacar que mientras menor sea ésta, mayor es el tamaño del objeto impreso. Por ejemplo, si una imagen de 815 pixeles de ancho por 210 pixeles de alto se imprime a 72 dpi ocupará 28.75 cm de ancho por 7.41 cm de alto, pero si la misma imagen se imprime a 300 dpi ocupará 6.9 cm por 1.68 cm, y si la misma imagen se imprime a 600 dpi ocupará 3.45 cm por 0.89 cm que es la mitad en alto y ancho que la impresa a 300 dpi.

Además, existe una estrecha relación entre la resolución de impresión y la resolución del dispositivo de lectura. Los defectos que pueda causarle a una imagen cualquier imperfección en la impresión afecta en mayor cantidad cuanto menor sea la resolución de impresión, es decir, cuando más grande salga impreso el símbolo de código de barras. La explicación radica en que los glyphs impresos pueden llegar a ser interpretados por el dispositivo de lectura de manera errónea. Si la resolución del dispositivo de lectura es tan precisa que permite ver a los glyphs como si fueran manchas o blancos entonces creerá que hay más barras negras o espacios en blanco y por lo tanto estará leyendo información incorrecta. La resolución del dispositivo de lectura debe ser tal que los glyphs embebidos resulten imperceptibles para él. Luego, si la resolución de impresión es menor, el símbolo de código de barras con los glyphs embebidos se imprimirá en un tamaño mayor y los glyphs serán de mayor tamaño también, por lo que el dispositivo lector estará más proclive a detectar los glyph e interpretarlos como nuevas barras negras o espacios en blanco, y en ese caso fallará la lectura.

Respecto al caso 2, si bien la resolución es de 600 dpi, el símbolo de código de barras tiene las barras blancas completas de glyphs, y éstos además con demasiada altura (4 pixeles) y grosor (2 pixeles). Con esas tres características, el dispositivo de lectura (escáner) no puede determinar bien los comienzos o fines de barra.

Los otros casos que fallaron son el 9, 11, 15, 17, 19 y el 25. Todos ellos son impresiones a 300 dpi. A continuación se analizará caso por caso.

Caso 9. Grosor 2, Altura 4, Densidad 0 y Blancos Incompleto. Se diferencia del caso 1 por la completitud de los blancos. El problema más importante en este caso es la densidad de 300 dpi.

Caso 11. Grosor 2, Altura 3, Densidad 0 y Blancos Incompleto. Similar al caso 9, con la diferencia que algunas lecturas resultaron satisfactorias.

Caso 15. Grosor 1, Altura 3, Densidad 0 y Blancos Incompleto. Similar al caso 11, con menor grosor. De nuevo, el problema es la baja resolución de 300 dpi.

Caso 17. Grosor 2, Altura 4, Densidad 1 y Blancos Completo. Aquí la diferencia fundamental es la densidad. Con esta densidad, el dispositivo de lectura (escáner) logra identificar las barras negras. De todos modos en algunos casos la lectura resultó fallida.

Caso 19. Grosor 2, Altura 3, Densidad 1 y Blancos Completo. Similar al caso 17, pero con otra altura de glyph.

Caso 25. Grosor 2, Altura 4, Densidad 1 y Blancos Incompleto. Este caso, si bien no debería traer problemas ya que tiene poca densidad y blancos incompleto, la resolución de 300 dpi sigue siendo un problema.

Conclusiones de las pruebas de lectura
Dada la tecnología que se está usando y los resultados observados, se concluye que la disposición y características de los glyphs dentro del símbolo de código de barras está en relación directa con la resolución de impresión elegida, y por ello no siempre se podrá utilizar la misma disposición de glyphs (Densidad, Blancos Completos, etc) para el mismo símbolo de código de barras a diferentes resoluciones.

Si se lo analiza desde el punto de vista de la completitud de los Blancos, se obtuvieron los siguientes resultados:

- la cantidad de lecturas denominadas OK con Blancos Completos fue de 11/16
- la cantidad de lecturas denominadas OK más la cantidad de lecturas denominadas REGULAR con Blancos Completos fue de 13/16
- la cantidad de lecturas OK con Blancos Incompletos fue de 12/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Blancos Incompletos fue de 15/16

En todos los casos, se ha tomado el conjunto de pruebas con Blancos Completos e Incompletos en forma separada, y así se tiene un total de 16 casos para cada conjunto. Se concluye que en términos generales existe una leve mejoría cuando se utiliza la disposición de glyphs Blancos en forma Incompleta.

Desde el punto de vista de la Densidad se lograron los siguientes resultados:

- la cantidad de lecturas OK con Densidad 0 fue de 10/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Densidad 0 fue de 12/16
- la cantidad de lecturas OK con Densidad 1 fue de 13/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Densidad 1 fue de 16/16

Se deduce que en términos generales es mejor tomar Densidad 1. La desventaja evidente es la menor cantidad de información que se puede codificar como glyphs en el símbolo de código de barras.

Desde el punto de vista del Grosor se obtuvieron los siguientes resultados:

- la cantidad de lecturas OK con Grosor 2 fue de 8/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Grosor 2 fue de 12/16
- la cantidad de lecturas OK con Grosor 1 fue de 15/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Grosor 1 fue de 16/16

Se concluye que en términos generales es mejor tomar un Grosor de 1.

Desde el punto de vista de la Altura se obtuvieron los siguientes resultados:

- la cantidad de lecturas OK con Altura 3 fue de 12/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Altura 3 fue de 15/16
- la cantidad de lecturas OK con Altura 4 fue de 11/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Altura 4 fue de 13/16.

Se concluye que en términos generales es mejor utilizar una Altura de 3.

Desde el punto de vista de la Resolución de Impresión se obtuvieron los siguientes resultados:

- la cantidad de lecturas OK con Resolución 300 dpi fue de 8/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Resolución 300 dpi fue de 13/16
- la cantidad de lecturas OK con Resolución 600 dpi fue de 15/16
- la cantidad de lecturas OK más la cantidad de lecturas REGULAR con Resolución 600 dpi fue de 15/16.

Se concluye que en términos generales es mejor utilizar una Resolución de 600 dpi.

Si se desea lograr una buena lectura del símbolo de código de barras con una impresión realizada a 300 dpi debe utilizarse un valor de Grosor 1; el resto de las variables pueden combinarse libremente, salvo en el caso de Grosor 2, Altura 4 y Densidad 0. El caso de Grosor 2, Altura 3, Densidad 0 y Blancos Completos debe evitarse. La desventaja de esta configuración es que el glyph es muy poco grueso y por lo tanto complicará la posterior detección de glyphs.

Si en cambio, se desea lograr una buena lectura del símbolo de código de barras con una impresión realizada a 600 dpi el único requisito es que el valor Grosor sea 1. La única combinación de valores que no debe utilizarse es la de Grosor 2, Altura 4, Densidad 0 y Blancos Completos.

Luego de haber analizado todos los casos de prueba y los distintos valores en forma combinada, la configuración ideal general encontrada para una buena lectura del código de barras con glyphs embebidos es: Grosor 1, Altura 3, Densidad 1 y Blancos Incompletos. Sin embargo, también se puede utilizar valores combinados como Grosor 2, Altura 4 y Densidad 0 todos al mismo tiempo, siempre y cuando el valor Blancos Completos sea NO y la Resolución de Impresión sea de 600 dpi; la ventaja en este último caso es que, a pesar que se tienen más defectos en el símbolo de código de barras, esa combinación permite mayor poder expresivo y una mejor detección posterior de los glyphs.

Pruebas de Detección de glyphs

Para la detección se utilizara una configuración de blancos incompleto con densidad 0. Se irán variando los anchos de barra (5 y 10 pixeles), los anchos de celda (5, 8 y 10 pixeles) y el grosor del glyph (1, 2 y 3 pixeles).

Se realizaron distintos tipos de pruebas, con el siguiente mensaje embebido:

"Propuesta de Tesis: Codificación de información embedded en código de barras convencionales. Autores: Eduardo Raichman (LU 231/89), Hernán Podolsky (LU 659/88). Director: Eduardo Rodríguez"

Este mensaje generó a través de la función *dgencode* de las librerías de Xerox PARC el siguiente DataGlyphs:

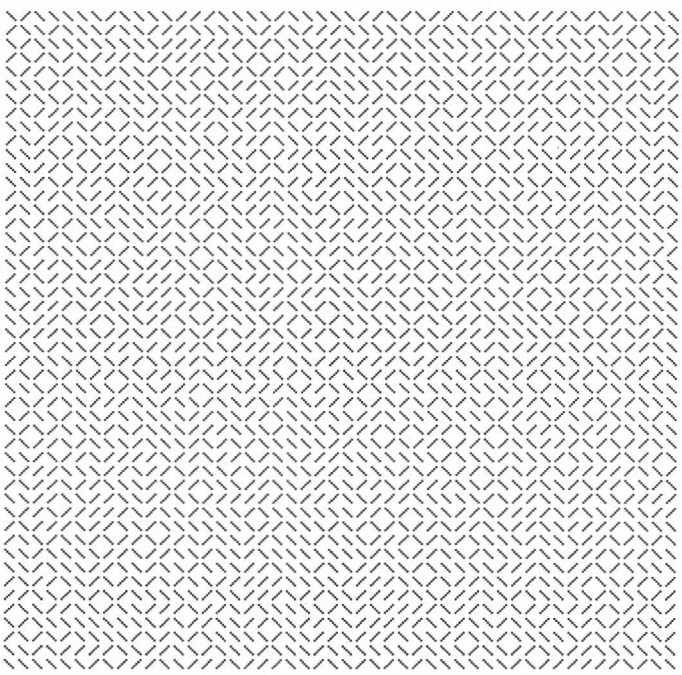


Fig. C-6. DataGlyph original, salida del *dgencode.*

El DataGlyph tiene una dimensión de 48 x 48 = 2304 glyphs

Con dicho mensaje de prueba y el DataGlyph resultante se realizaron las siguientes pruebas:

Prueba 1: Características del símbolo de código de barras y los glyphs.
Tipo: Code39
Blancos Completo: NO
Grosor del glyph: 2
Altura del glyph: 7
Ancho de celda: 8
Ancho de barra angosta: 10
Densidad: 0
Resolución de Impresión: 300 dpi
Resolución del Dispositivo de Lectura: 600 dpi

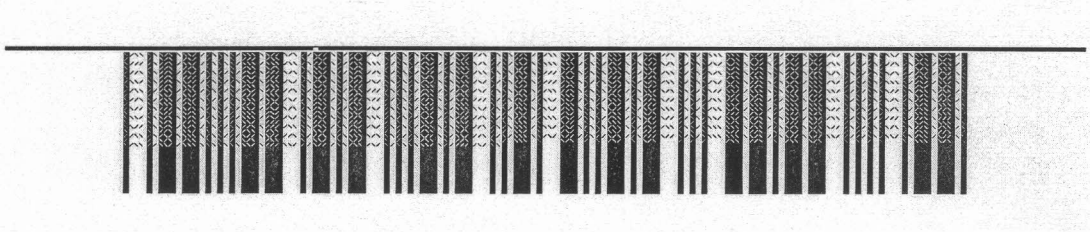


Fig. C-7. Imagen escaneada de la Prueba 1.

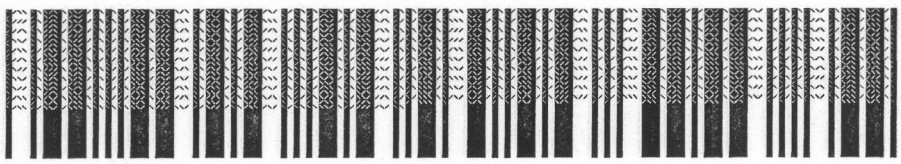


Fig. C-8. Imagen cuantizada de la Prueba 1.

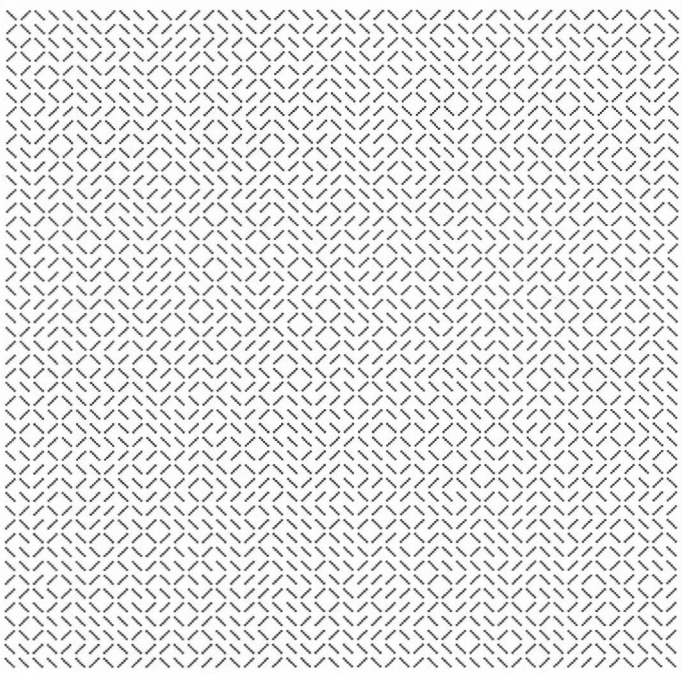


Fig. C-9. DataGlyph resultante de la Prueba 1.

Cantidad de errores: 0. La decodificación resultó exitosa.

Prueba 2: Características del símbolo de código de barras y los glyphs.
Tipo: Code39
Blancos Completo: NO
Grosor del glyph: 2
Altura del glyph: 6
Ancho de celda: 8
Ancho de barra angosta: 10
Densidad: 0
Resolución de Impresión: 300 dpi
Resolución de Escaneo: 600 dpi

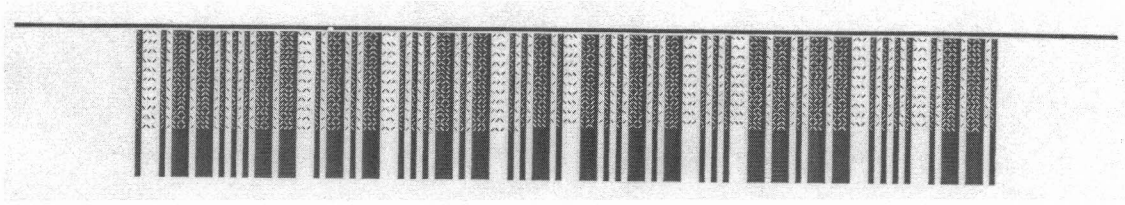


Fig. C-10. Imagen escaneada de la Prueba 2.

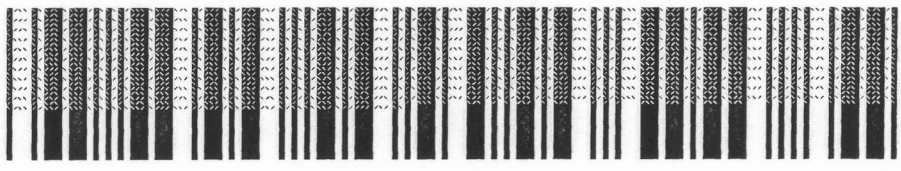


Fig. C-11. Imagen cuantizada de la Prueba 2.

Matriz resultante: Igual a la prueba 1.

Cantidad de errores: 0. La decodificación resultó exitosa.

Prueba 3: Características del símbolo de código de barras y los glyphs.
Tipo: Code39
Blancos Completo: NO
Grosor del glyph: 2
Altura del glyph: 4
Ancho de celda: 5
Ancho de barra angosta: 5
Densidad: 0
Resolución de Impresión: 300 dpi
Resolución de Escaneo: 600 dpi

Fig. C-12. Imagen escaneada de la Prueba 3.

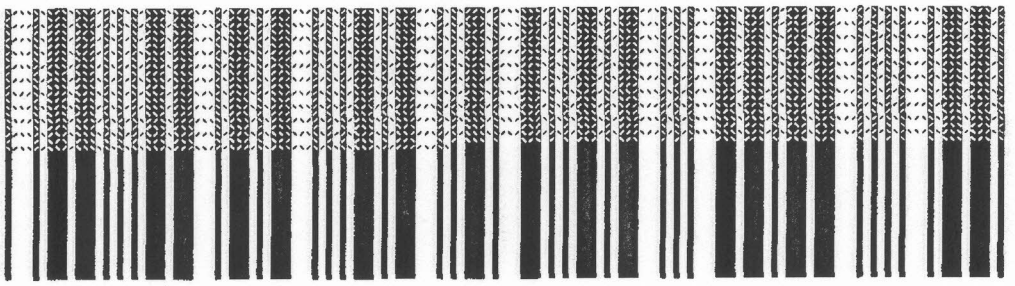


Fig. C-13. Imagen cuantizada de la Prueba 3.

Matriz resultante: Igual a la prueba 1.

Cantidad de errores: 0. La decodificación resultó exitosa.

Prueba 4: Características del símbolo de código de barras y los glyphs.
Tipo: Code39
Blancos Completo: NO
Grosor del glyph: 2
Altura del glyph: 3
Ancho de celda: 5
Ancho de barra angosta: 5
Densidad: 0
Resolución de Impresión: 300 dpi
Resolución de Escaneo: 600 dpi

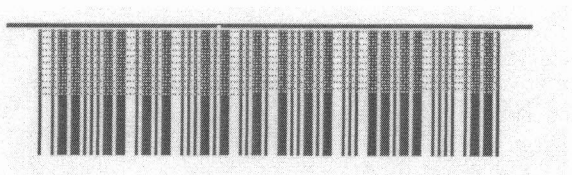


Fig. C-14. Imagen escaneada de la Prueba 4.

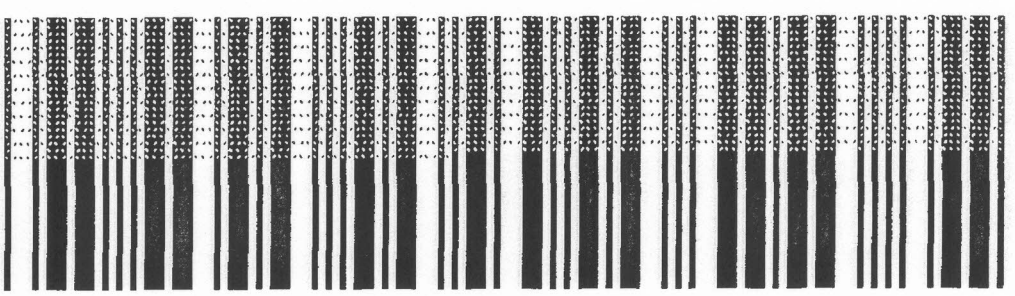
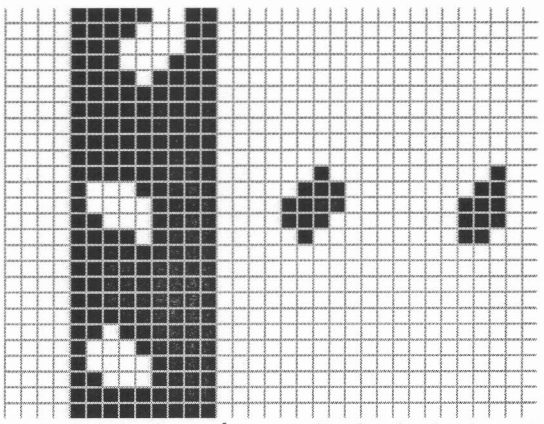


Fig. C-15. Imagen cuantizada de la Prueba 4.

La decodificación NO resultó exitosa.

El problema en esta prueba es que los glyphs no son lo suficientemente altos como para poder ser detectados correctamente. Lo ideal es que sean más altos que anchos.

Véase el extracto de la imagen ampliada en la Fig. C-16:



Obsérvese la forma proporcional en altura y grosor de los glyphs. Lo ideal sería un glyph más alto que ancho.

Fig. C-16. Porción ampliada de la Fig. C-15.

Prueba 5: Características del símbolo de código de barras y los glyphs.
Tipo: Code39
Blancos Completo: NO
Grosor del glyph: 1
Altura del glyph: 4
Ancho de celda: 5
Ancho de barra angosta: 5
Densidad: 0
Resolución de Impresión: 300 dpi
Resolución de Escaneo: 600 dpi

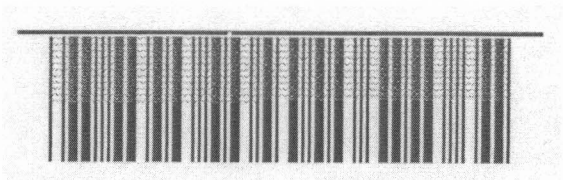


Fig. C-17. Imagen escaneada de la Prueba 5.

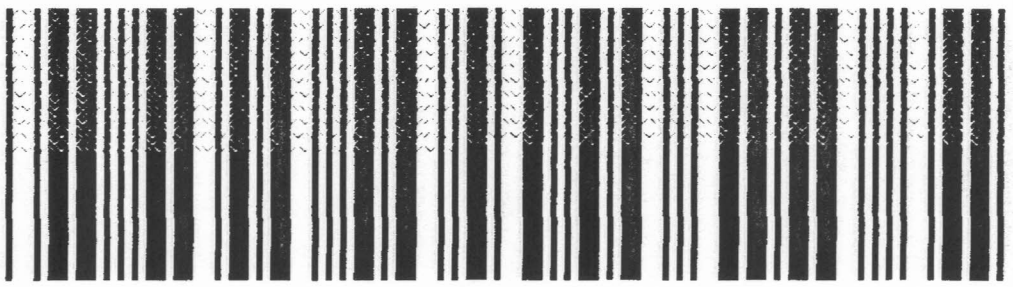
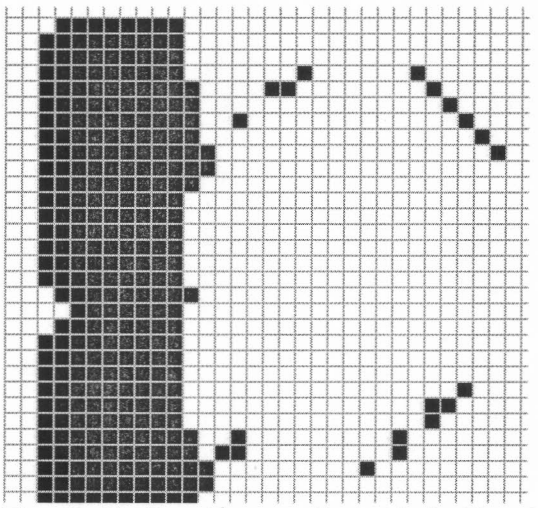


Fig. C-18. Imagen cuantizada de la Prueba 5.

La decodificación NO resultó exitosa.

El problema de esta prueba es que se está utilizando un grosor de glyph muy angosto para los glyphs, y el dispositivo de lectura utilizado no alcanza a escanear con suficiente precisión la imagen. Se debería utilizar un escaner más preciso y por lo tanto un mayor costo de hardware. La solución es utilizar un grosor adecuado como el de la prueba 3.

Véase el extracto de la imagen ampliada en la siguiente figura:



En la barra negra deberían aparecer glyphs, pero no se ven debido a la baja calidad del escaneo.

Fig. C-19. Porción ampliada de la Fig. C-18.

Prueba 6: Características del símbolo de código de barras y los glyphs.
Tipo: Code39
Blancos Completo: NO
Grosor del glyph: 1
Altura del glyph: 7
Ancho de celda: 8
Ancho de barra angosta: 10
Densidad: 0
Resolución de Impresión: 300 dpi
Resolución de Escaneo: 600 dpi

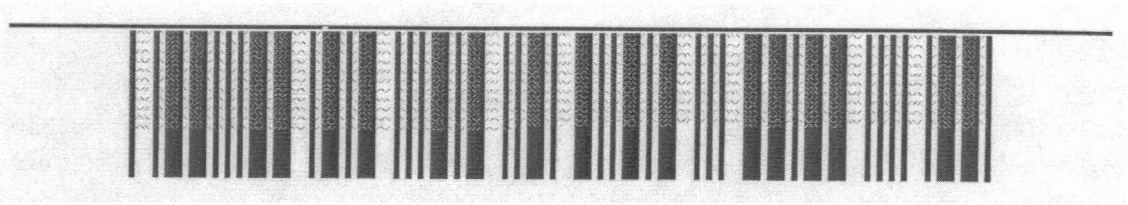


Fig. C-20. Imagen escaneada de la Prueba 6.

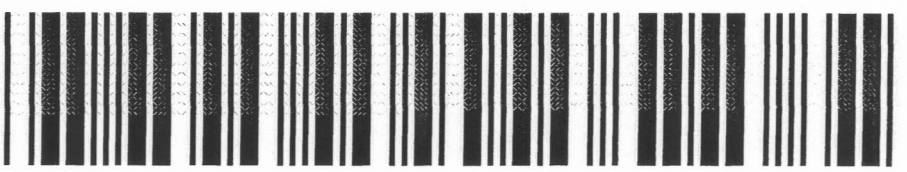


Fig. C-21. Imagen cuantizada de la Prueba 6.

La decodificación NO resultó exitosa.

El problema es el mismo que se describió para el caso anterior.

Análisis y conclusiones de las pruebas de detección de glyphs
La detección de glyphs es un proceso que utiliza técnicas de procesamiento de imágenes. Por lo tanto, mientras más nítida sea la imagen mayor probabilidad habrá que la decodificación sea exitosa.

Algunas pruebas expuestas en este apéndice no han resultado exitosas debido a que los glyphs resultaban demasiado angostos o poco altos, por lo que la imagen perdía la información al ser escaneada. Una mejor tecnología de escaneo hubiese entregado resultados superiores y el proceso de *extracción* de glyphs hubiese trabajado con éxito. Si no se puede contar con la mejor tecnología de escaneo, un uso adecuado del ancho y la altura del glyph será la solución a este problema.

Apéndice D

Ejemplos de cálculo de eficiencia de los glyphs

Ejemplo 1
Mensaje: "1234 abcd"
Simbología: Code 128
Ancho de módulo utilizado: 5, 6, 7 y 8

Como se vio en la Tabla A-2, para un mensaje "1234 abcd" utilizando simbología Code 128, suponiendo anchos de módulo 5, 6, 7 y 8 se llegó a que los mejores tamaños de celda para cada caso son: 5, 6, 5 y 5 respectivamente. La Tabla D-1 muestra un resumen de los resultados de la Tabla A-2:

Tabla D-1. Resultados de la tabla A-2.

Valores	*Ancho de módulo X*			
	5 pixeles	*6 pixeles*	*7 pixeles*	*8 pixeles*
Cantidad máxima de glyphs por fila	123	123	137	165
Desperdicio	0	0	176	159
Tamaño de celda óptima	5	6	5	5
Longitud en pixeles	615	738	861	984

Como se explicó en la Sección 4.2 del Capítulo 4, el proceso *merge* coloca glyphs en las barras negras y en las barras blancas sólo en las filas impares, y sólo en las barras negras en las filas pares. La Tabla D-2 muestra para cada valor de X en este ejemplo, cuantos glyphs pueden colocarse en las filas pares:

Tabla D-2. Cantidad de glyphs en filas pares

Medida	*X=5, C=5*	*X=6, C=6*	*X=7, C=5*	*X=8, C=5*
Cantidad de glyphs	60	60	66	80

Para Code 128, la altura de un símbolo es de un mínimo del 15 % de la longitud. [Pal91] Por ejemplo, el símbolo que tiene un tamaño de celda de 5 pixeles la altura es de 92,25 pixeles, es decir, de 18,45 módulos, por lo que serán utilizados 18 módulos. Para cada ancho de módulo se mantiene constante la longitud total del símbolo, pero cambia la cantidad de glyphs que se pueden expresar según cambia el tamaño de la celda.

En la Tabla D-3 se pueden ver los tamaños de longitud, altura y área medidos en módulos para cada caso de este ejemplo:

Tabla D-3. Longitud, altura y área en módulos.

Medida	*X=5*	*X=6*	*X=7*	*X=8*
Longitud	123	123	123	123
Altura	18	18	18	18
Area	2214	2214	2214	2214

La Tabla D-4 muestra la altura en pixeles y la cantidad de glyphs que pueden expresarse en cada columna, en cada caso de este ejemplo:

Tabla D-4. Altura en pixeles y cantidad de glyphs en altura.

Medida	X=5, C=5	X=6, C=6	X=7, C=5	X=8, C=5
Altura	90	108	126	144
Cantidad de glyphs en altura	18	18	25	28

La Tabla D-5 muestra, la cantidad de glyphs que pueden expresarse en total, calculando la cantidad de filas pares e impares, luego la cantidad de glyphs que pueden expresarse en cada una de ellas, para cada caso de este ejemplo:

Tabla D-5. Totales de glyphs.

Medida	X=5, C=5	X=6, C=6	X=7, C=5	X=8, C=5
Cantidad de filas impares	9	9	13	14
Cantidad de filas pares	9	9	12	14
Total de glyph en filas impares	1107	1107	1781	2310
Total de glyph en filas pares	540	540	792	1120
Total glyphs en el símbolo	1647	1647	2573	3430

Por último, la Tabla D-6 muestra los resultados de las medidas de información para cada caso de este ejemplo:

Tabla D-6. Medidas de Información.

Medida de la Información	X = 5, C = 5	X = 6, C = 6	X = 7, C = 5	X = 8, C = 5
Cantidad de bits de información	1647	1647	2573	3430
Eficiencia de área en %	74,39	74,39	116,21	154,92
Densidad en bits por pixel	0,0298	0,0207	0,0237	0,0242

Ejemplo 2
Mensaje: genérico de 9 caracteres de datos
Simbología: Code 128
Ancho de módulo utilizado: 5, 6, 7 y 8

La Tabla D-7 muestra los resultados obtenidos para cada caso de este ejemplo:

Tabla D-7. Resultados obtenidos para el Ejemplo 2.

Medida	X=5, C=5	X=6, C=6	X=7, C=5	X=8, C=5
Altura en módulos	20	20	20	20
Area en módulos	2680	2680	2680	2680
Altura en pixeles	100	120	140	160
Total de glyphs en fila impar	134	134	148	181
Total de glyphs en fila par	68	68	77	90
Total de glyphs en altura	20	20	28	32
Cantidad de filas impares	10	10	14	16
Cantidad de filas pares	10	10	14	16
Total de glyphs en filas impares	1340	1340	2072	2896
Total de glyphs en filas pares	680	680	1078	1440
Total de glyphs en el símbolo	2020	2020	3150	4336
Medida de la Información	**X=5, C=5**	**X=6, C=6**	**X=7, C=5**	**X=8, C=5**
Cantidad de bits de información	2020	2020	3150	4336
Eficiencia de área en %	75,37	75,37	117,54	161,79
Densidad en bits por pixel	0,0301	0,0209	0,0240	0,0253

Ejemplo 3
Mensaje: genérico, de 9 caracteres de datos
Simbología: Code 39 (utiliza un módulo entre caracteres)
Ancho de módulo utilizado: 5, 6, 7 y 8

La Tabla D-8 muestra los resultados obtenidos para cada caso de este ejemplo:

Tabla D-8. Resultados obtenidos para el Ejemplo 3.

Medida	***X=5, C=5***	***X=6, C=6***	***X=7, C=5***	***X=8, C=5***
Altura en módulos	21	21	21	21
Area en módulos	2982	2982	2982	2982
Altura en pixeles	105	126	147	168
Total de glyphs en fila impar	142	142	142	175
Total de glyphs en fila par	65	65	65	80
Total de glyphs en altura	21	21	21	33
Cantidad de filas impares	11	11	11	17
Cantidad de filas pares	10	10	10	16
Total de glyphs en filas impares	1562	1562	1562	2975
Total de glyphs en filas pares	650	650	650	1280
Total de glyphs en el símbolo	2212	2212	2212	4255
Medida de la Información	***X=5, C=5***	***X=6, C=6***	***X=7, C=5***	***X=8, C=5***
Cantidad de bits de información	2212	2212	2212	4255
Eficiencia de área en %	74,18	74,18	74,18	142,69
Densidad en bits por pixel	0,0297	0,0206	0,0151	0,0223

A partir de estos resultados, se observa que la cantidad de bits de información que permite almacenar un símbolo de código de barras de una dimensión sin glyphs es claramente inferior a la que en el mismo símbolo se puede almacenar con glyphs.

La Tabla D-9 muestra los resultados comparativos de las Medidas de Información:

Tabla D-9. Comparación de Resultados. Simbologías con y sin glyphs.

Simbología - Ejemplo	***Sin glyphs***			***Con glyphs (promedio)***		
	Cantidad de bits de Información	***Eficiencia de área***	***Densidad (bits x pixel) (promedio)***	***Cantidad de bits de Información***	***Eficiencia de área***	***Densidad (bits x pixel)***
Code 128 – Mensaje "1234 abcd"	64	2,89%	0,0008	2324	104,98%	0,0296
Code 128 – Mensaje genérico	72	2,69%	0,0007	2881	107,52%	0,0251
Code 39 – Mensaje genérico	72	2,41%	0,0006	3630	91,30%	0,0219

Apéndice E

Observaciones de la Implementación

El número 65

Dentro de los detalles de implementación que tienen que ver con la barra de control, surge un curioso y arbitrario número 65 que se utiliza para colocar una muesca en la barra de control, que señala el tamaño de la celda de glyphs.

Existen ciertos cálculos que se realizan para posicionarse en barras o glyphs, y para tomar alguna información de control. Se realizan promediando un conjunto de valores, y mientras más valores formen parte de ese promedio menor será el error respecto a la posición real. Uno de esos cálculos se realiza para ubicar y recuperar la muesca en la barra de control.

La idea es tomar un número suficientemente grande, pero que sea menor que la longitud total de la barra de control. Durante el circuito de VUELTA se necesita procesar un mapa de bits que representa al símbolo de código de barras con glyphs. El tamaño de dicho mapa de bits difícilmente coincida con el del mapa de bits original del circuito de IDA. Si se toma la muesca del mapa de bits del circuito de VUELTA, y se sabe que está a una distancia 65*W, con sólo dividir por 65 se sabrá el valor de W, que es el ancho de la celda.

Si se toma un valor más chico puede ocurrir que en la imagen escaneada no todos los módulos sean del mismo tamaño (X). Puede haber diferencias de 1 pixel en el ancho de los módulos por lo que lo más conveniente es tomar la mayor cantidad de módulos posible para que el promedio sea el valor de ancho de celda más aproximado al real.

Se ha visto en distintas simbologías que la codificación de 1 solo caracter no alcanza ya que la barra de control será más corta que 65*W, sin embargo es extremadamente raro que se utilice una simbología de código de barras para codificar 1 solo caracter de información.

Restricciones de la aplicación

Algunas simbologías, además de contener barras y espacios, contienen en su parte inferior el mensaje a codificar en caracteres legibles por un humano. En estos casos, los algoritmos desarrollados producirán resultados anómalos ya que no podrán calcular bien la altura de las barras ni la cantidad de glyphs que pueden dibujarse en el símbolo de código de barras.

Otro problema no resuelto por los algoritmos desarrollados es el de la mayor longitud de las barras de los caracteres de arranque, parada e intermedio en simbologías como UPC y EAN. Los efectos que producirán son los mismos que se mencionaron para el caso anterior. (Ver Fig. E-1)

Fig E-1. Símbolo de Código de Barras UPC

Para solucionar los dos problemas mencionados habría que observar previamente el símbolo en forma global, eliminar los números, recortar las barras, y luego ejecutar el algoritmo tal como está desarrollado en este momento. Otra forma sería eliminar sólo los números, manteniendo la altura original de las barras, y acondicionando el algoritmo de codificación para poder aprovechar la altura extra tanto en las barras como en los espacios.

Otra restricción de los algoritmos es que la utilización de la barra de control no es un estándar de ninguna simbología de código de barras. Para solucionar este problema se deberá implementar algún otro mecanismo de rotación que no use guías, que a su vez pueda detectar cual lado es el superior y cual el inferior tan sólo con la información que tiene el símbolo. También será necesario un algoritmo sofisticado que pueda determinar con precisión el tamaño de la celda.

Referencias

- [Abr86] Norman Abramson, "*Teoría de la Información y Codificación*", International Thonson Editores Spain Paraninfo S.A., 1986
- [Ham50] R. W. Hamming, "*Error Detecting and Correcting Code*", The Bell System Techincal Journal vol 26 Nº 2, pp 147- 160 Abril 1950
- [Hec01] David L. Hecht, "*Printed Embedded Data Graphical User Interfaces*", Computer, vol 34, Nº 3 Marzo 2001, pp 47-55
- [Pal91] Roger Palmer, "*The Barcode Book*", Helmers Publishing Inc, 1991
- [Pav90] Theo Pavlidis, Jerome Swartz and Ynjiun P. Wang, "*Fundamentals of bar code information theory*" Computer, 23(4):74—86 ,Abril 1990
- [Pav92] Theo Pavlidis, Jerome Swartz and Ynjiun P. Wang, "*Information Encoding with Two-Dimensional Barcodes*", Computer, Junio 1992, pp 18-28
- [WWW1] Xerox PARC, "*www.dataglyphs.com*"
- [WWW2] "*www.basics.ie/History.htm*". Resumen de: T. Seideman, "*The History of Barcodes*", American Heritage of Invention and Technology.
- [WWW3] "*www.adams1.com*"
- [WWW4] "*www.barcodemill.com*"
- [WWW5] "*www.codigo.com.ar*", el sitio de la organización EAN Argentina
- [WWW6] "*www.symbol.com*", el sitio de la Compañía Symbol Technologies
- [WWW7] "*www.fsiautomation.com*", el sitio de la Compañía FSI Automation
- [WWW8] "*www.idautomation.com*", el sitio de la Compañía ID Automation
- [WWW9] " *www2.xerox.com/xsis/dataglph.htm* ", el sitio de DataGlylphs en XSIS
- [WWW10] "*www.alphaave.com*", un sitio de promoción de tecnologías, propiedad de Xerox y del IT Lab del Instituto Tecnológico de Rochester