ÍNDICES DE COLECCIONES DE NUBES DE PUNTOS BAJO TRANSFORMACIONES DE SEMEJANZA.

TESIS

QUE PARA OBTENER EL TÍTULO DE
LIC. EN CIENCIAS FÍSICO-MATEMÁTICAS
PRESENTA

JONATAN SOFFER HERNÁNDEZ

ASESORES:
DR. EDGAR LEONEL CHÁVEZ GONZÁLEZ
DR. JORGE LUIS LÓPEZ LÓPEZ

FACULTAD DE CIENCIAS FÍSICO-MATEMÁTICAS MAT. LUIS MANUEL RIVERA GUTIÉRREZ

UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

MORELIA, MICHOACÁN
NOVIEMBRE DE 2020

ÍNDICE

LISTA DE FIGURAS

RESUMEN

Las consultas de nubes de puntos tienen amplias aplicaciones en áreas como geografía, astronomía, y en general en cualquiera en la que objetos de interés puedan ser representados como puntos en el espacio.
En esta tesis se presentan dos métodos para realizar consultas sobre nubes de puntos bidimensionales utilizando invariantes hiperbólicas derivadas de la estructura de estas. El primero realiza consultas para responder si dada una nube de puntos bidimensional existe alguna transformación de semejanza tal que la imagen de la nube bajo esta se encuentra dentro de una colección de nubes de puntos fijada de antemano. El segundo realiza consultas para responder si dada una nube de puntos bidimensional existe una transformación de semejanza tal que la imagen de la nube bajo esta sea un subconjunto de una nube de puntos fijada de antemano. Finalmente se muestran resultados de pruebas empíricas para medir el desempeño de estos métodos.

ABSTRACT

Point cloud queries have a wide range of applications in areas like geography, astronomy, and in general in any area where objects of interest can be represented as points in space.
In this thesis are presented two methods to perform queries over bidimensional point clouds using hyperbolic invariants derived from their structure. The first one performs queries to answer if given a bidimensional point cloud there is a similarity transformation such that the image of the cloud under it is in a collection of point clouds fixed beforehand. The second one performs queries to answer if given a bidimensional point cloud there is a similarity transformation such that the image of the cloud under it is a subset of a point cloud fixed beforehand. Finally, results of empiric tests to measure the performance of these methods are shown.

Palabras Clave— consultas de nubes de puntos, invariantes hiperbólicas, reconocimiento de patrones, consultas de constelación, emparejamiento bajo semejanzas

1 INTRODUCCIÓN

El encontrar colecciones de objetos que tengan alguna relación métrica de interés entre sí tiene varias aplicaciones en astronomía, geología, y diseño, por nombrar algunas áreas. Esta tesis se enfoca en la variante del problema de emparejar un patrón de puntos bidimensionales a *todos* los subconjuntos de puntos en una base de datos a diferentes escalas.
Consideremos el siguiente caso de uso: dados puntos que describen un patrón conocido, digamos un hexágono regular, encontrar conjuntos de estrellas que formen un hexágono regular a todas las escalas permitiendo ruido aditivo. El extraer *patrones de puntos* o *constelaciones* involucra emparejar un patrón de consulta a conjuntos de puntos, de manera tal que cada uno de estos conjuntos obedezca las restricciones geométricas expresadas por el patrón de consulta de acuerdo a cierto criterio de emparejamiento. Consideramos tres criterios de emparejamiento en orden creciente de generalidad. Se comienza con una discusión de las *consultas de constelación puras* en las cuales los conjuntos de puntos respuesta deben emparejar exactamente las distancias del patrón de consulta hasta un error aditivo. Luego, se tratan las *consultas isotrópicas*, en las cuales los puntos respuesta emparejan al patrón de consulta hasta un factor de escalamiento y un error aditivo. Finalmente se tratan las *consultas de constelación no isotrópicas* en las cuales al menos un subconjunto de las distancias puede ser estirado hasta cierto punto. El ejemplo recurrente de este capítulo provendrá de la astronomía, pero los resultados son igualmente aplicables para otras áreas.

1.1 FORMULACIÓN DEL PROBLEMA

Un Dataset D se define como un conjunto de elementos que tienen coordenadas en algún espacio n-dimensional. Un patrón constelación $Q_k = \{q_1, q_2, \ldots, q_k\}$, es un conjunto de k elementos en ciertas ubicaciones y un factor de error ϵ. Sin pérdida de generalidad, trataremos a los elementos de D como vectores en algún espacio n-dimensional.
Como fue presentado de manera informal en el párrafo anterior, se definen tres clases de consultas de constelación, en nivel creciente de generalidad:

1. Consulta de constelación pura.
2. Consulta de constelación isotrópica.
3. Consulta de constelación no isotrópica.

Las soluciones a consultas de constelación puras consisten de conjuntos de k elementos, cuyas distancias a pares emparejan exactamente a las de la consulta dentro de un error aditivo ϵ.
Un conjunto $S = \{s_1, \ldots, s_k\} \subseteq D$ *empareja puramente* al patrón de consulta $Q = \{q_1, \ldots, q_k\}$ si, para todas $1 \leq i, j \leq k$,
$\left\|q_i - q_j\right\| - \epsilon \leq \left\|s_i - s_j\right\| \leq \left\|q_i - q_j\right\| + \epsilon$. Dado que el emparejamiento depende únicamente de las distancias, las consultas de constelación puras son invariantes bajo rotaciones y traslaciones.

Las consultas de constelación isotrópicas admiten cualquier solución que sea admisible para las consultas puras (dentro de un factor de error aditivo) con la libertad adicional de que las distancias pueden ser multiplicadas uniformemente por un valor arbitrario. Esto es, además de ser invariantes bajo rotaciones y traslaciones, las consultas isotrópicas son invariantes bajo escalamientos. Un conjunto $S = \{s_1, \ldots, s_k\} \subseteq D$ *empareja isotrópicamente* a una consulta $Q = \{q_1, \ldots, q_k\}$ si existe un factor de escala f tal que para todas $1 \leq i, j \leq k$, $f \times \left\|q_i - q_j\right\| - \epsilon \leq \left\|s_i - s_j\right\| \leq f \times \left\|q_i - q_j\right\| + \epsilon$. El reto en este caso es encontrar cada emparejamiento aunque f pueda tomar cualquier valor arbitrario.
Finalmente, las consultas de constelación no isotrópicas admiten deformaciones controladas en al menos una de las distancias a pares. Esto es, un conjunto $S = \{s_1, \ldots, s_k\} \subseteq D$ *empareja no isotrópicamente* a un patrón de consulta $Q = \{q_1, \ldots, q_k\}$ si existe algún factor de escala f y algún conjunto de modificadores del factor de escala $\{m_{i,j}\}$, con $m_{i,j} \geq 1$ tal que $f \times \left\|q_i - q_j\right\| - \epsilon \leq \left\|s_i - s_j\right\| \leq (f \times m_{i,j} \times \left\|q_i - q_j\right\|) + \epsilon$. Aquí el reto adicional es que cada factor de estirado puede tomar cualquier valor entre 1 y $m_{i,j}$. De nuevo, se desea encontrar a todos los apareamientos tales para cada factor de escalado f y factor de estirado de hasta $m_{i,j}$.

1.2 RESUMEN DE LOS CAPÍTULOS

Este es un breve esquema del contenido de cada capítulo. En el capítulo 2 se presenta el problema de las consultas de constelación. En el capítulo 3 se presenta se presentan invariantes afines de polígonos generalizados y su aplicación para emparejamientos de estos. En el capítulo 4 se presentan métodos para realizar consultas de nubes de puntos. En el capítulo 5 se muestran los resultados experimentales. Finalmente, en el capítulo 6 se presentan las conclusiones.

2 CONSULTAS DE CONSTELACIÓN

2.1 INTRODUCCIÓN

En este capítulo se revisan las soluciones a los problemas de consulta de constelación propuestos por [7], basados en una implementación distribuida usando *Apache Spark*.

2.2 CONSULTAS DE CONSTELACIÓN PURAS

Responder *consultas de constelación puras* sobre datos astronómicos requiere técnicas eficientes de procesamiento de consultas ya que el catálogo podría contener miles de millones de objetos celestes. La emisión número 14 de los datos Sloan *Digital Sky Survey* (SDSS), por ejemplo, contiene aproximadamente 1.3×10^9 objetos, lo cual llevaría a la evaluación de aproximadamente $\frac{1.3\times(10^9)^4}{4!} = 1.2 \times 10^{35}$ conjuntos candidatos para una consulta de tamaño 4. (Nótese que el tamaño del conjunto solución podría ser de ese orden de magnitud. Por ejemplo, $1/4$ de las estrellas podría estar en alguna de las 4 en esquinas de algún cuadrado). Algoritmos de *point set registration*, como *Iterative Closest Point* (ICP) pueden hacerlo en tiempo $O(mn)$, donde n es el número de elementos en la consulta Q y m es el tamaño del conjunto de datos D. Sin embargo, ICP requiere que los dos conjuntos de puntos sean de aproximadamente el mismo tamaño, y se encuentren cerca el uno del otro, lo cual es muy restrictivo para la búsqueda de patrones de puntos. Entonces, para volver el problema computacionalmente manejable, cuando $\binom{|D|}{k}$ es grande, la estrategia presentada por [7] para procesar consultas de constelación puras involucra tres técnicas principales. Primero, se usa un *quadtree* para recuperar a aquellas estrellas que se encuentran a lo más a cierta distancia de cada estrella. Segundo, se usan propiedades de los elementos de la consulta (como brillo y frecuencia en el contexto astronómico) para restringir conjuntos de vecinos candidatos. Tercero, para cada estrella y sus vecinos candidatos se resuelve un *join* espacial de $k-1$ estrellas combinado con un *bucketed spatial join* de $k-2$ estrellas. Por ejemplo, para buscar un triángulo con lados de longitudes L_1, L_2, y L_3, se quiere encontrar puntos a, b, c tales que $L_1 - \epsilon \leq \|a-b\| \leq L_1 + \epsilon$, $L_2 - \epsilon \leq \|b-c\| \leq L_2 + \epsilon$, y $L_3 - \epsilon \leq \|c-a\| \leq L_3 + \epsilon$.
Las secciones siguientes describen las técnicas de procesamiento de consultas a detalle.

2.2.1 *Particionamiento de Datos*

De acuerdo al método propuesto por [7] el contestar consultas de constelación puras involucra emparejar a cada estrella en el catálogo contra todas las estrellas vecinas que se encuentren a una distancia apropiada de acuerdo a la consulta, un procedimiento costoso en catálogos grandes. Para reducir este costo y hacer más eficiente el despliegue en *Spark*, se particiona

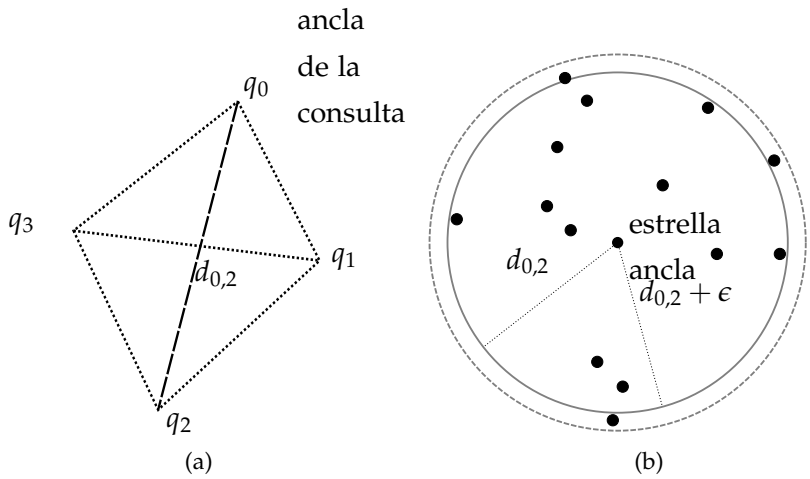
el conjunto de datos de manera *off-line* en un paso de pre-proceso. El particionamiento de los datos aplica un algoritmo basado en un histograma de equi-profundidad sobre una de las dimensiones espaciales.
La salida del procedimiento de particionado reparte al conjunto de datos en p particiones primarias. En el caso astronómico esto produce anillos alrededor de la Tierra. Cada partición primaria se extiende con un conjunto vecindario que contiene objetos en ambas particiones vecinas separados a lo más *max-distance* de los bordes. *max-distance* representa al radio máximo desde el *ancla* (intuitivamente, el centroide mediano) de cualquier consulta de interés. Para la astronomía, la distancia máxima ha sido definida como 5, 10 y 15 arcosegundos (0.0041 grados), basándose en retroalimentación de astrónomos. El ancho mínimo de cada partición es *max-distance* $+ \epsilon$, para que cada estrella en la partición primaria que pueda ser potencialmente mapeada al ancla de la consulta encuentre a todas las estrellas relevantes en la partición primaria misma, o en alguna de las dos particiones vecinas.
El particionado permite la paralelización del procedimiento de búsqueda dado que todas las estrellas de una partición pueden ser probadas como emparejamientos del ancla de la consulta de manera paralela con todas las estrellas de todas las demás particiones.

2.2.2 *Indexando Espacios de Datos*

Para indexar el espacio de datos se utiliza un *quadtree* como estructura de datos.

Definición 1. *Para un conjunto S de n puntos en $\mathcal{R}^2$, un quadtree es un árbol de grado 4, en el que cada uno de sus nodos v es asociado a un cuadrado R_v. R_v es particionado en cuantro cuadrados de tamaños iguales, cada uno de los cuales se asocia a un nodo hijo de V. Los cuadrados son particionados hasta que se cumpla alguan condición previamente definida, como la cantidad mázima de puntos en una región o una altura máxima del árbol. Los nodos que tienen asociados cuadrados que nocontienen más subdivisiones son las hojas del quadtree.*

[9]
El método de [7] se paraleliza entre las particiones, entonces se explica lo que sucede dentro de cada una. El espacio de datos dentro de una partición incluye a una primaria y a dos particiones vecinas (para el caso bidimensional). Para reducir el tiempo de búsqueda se construye un índice *quadtree* que cubre por completo el espacio de datos de la partición. (Por lo tanto, los *quadtrees* de particiones vecinas incluirán a algunas de las mismas estrellas). El proceso de construcción del *quadtree* [7] construye un árbol tal que los nodos hoja mantienen la siguiente propiedad: cualesquiera dos estrellas cubiertas por un cuadrante correspondiente a un nodo hoja no aparecen ambas en una misma solución. Esto se consigue especificando la altura del árbol como función de la distancia menor entre elementos de la consulta.
Elaborando sobre la intuición presentada posteriormente, el *ancla* de una consulta es un punto de la consulta cuya distancia máxima a cualquier punto de la consulta es minimal. Si existen varios puntos tales, cualquiera funciona.
Ahora, para cada hoja L del *quadtree* (hojas distintas pueden ser tratadas en paralelo), se encuentra a todas las demás hojas que podrían contener estrellas relevantes basándose en el tamaño de L y en la máxima distancia



(a) (b)

Figura 2.1: 2.1a Consulta de constalción pura con ancla y distancia máxima. 2.1b Elementos vecinos del representante del ancla.

desde el ancla hasta cualquier otro punto en el patrón de consulta. Ahora, considérese a cada estrella en L como un emparejamiento potencial a al ancla de la consulta. Como ejemplo considérese el caso ilustrado en la figura 2.1, que se describe en la subsección siguiente.

2.2.3 *Paso de Filtrado*

El paso de filtrado del vecindario se ilustra en la figura 2.1, en 2.1a una consulta Q tiene por elemento ancla a q_0 y *max-distance* corresponde a la mayor de las distancias ρ a los demás elementos de la consulta $d_{0,2}$. En 2.1b, se escoge a una estrella de la hoja actual L (a la cual se le llamará representante de ancla) como emparejamiento potencial del ancla, y a las estrellas dentro de una distancia de $d_{0,2} + \epsilon$ se les considera como candidatos preliminares para el apareamiento de distancia.

2.2.4 *K-1 Spatial Join*

Como se hizo notar anteriormente, el ancla de la consulta q_0 es el punto más cercano a todos los puntos de la consulta (cuya distancia máxima a todos los demás puntos es mínima). Entonces, para cada elemento del patrón $q_i, i \neq 0$, se determina si la distancia entre una estrella vecina s' y el representante de ancla s cumple $\|q_0 - q_1\| - \epsilon \leq \|s - s'\| \leq \|q_0 - q_1\| + \epsilon$. De ser así, se pone a s' en la *bucket* B_i, para $1 \leq i \leq k - 1$. Nótese que s' podría ponerse en varias *buckets*.

2.2.5 *Algoritmos de composición*

Recapitulando, cada *bucket* B_i es

$$B_i = \{s' \in S : (\|q_0 - q_i\| - \epsilon) \leq \left\|s' - s\right\| \leq (\|q_0 - q_i\| + \epsilon)\}$$

donde s es la estrella candidato a ancla, q_0 es la estrella ancla en el patrón de consulta, y q_i es la i-ésima estrella en el patrón de consulta, con $i \neq 0$.

2.2.5.1 *Spatial Bucket Join*

La manera más directa de encontrar conjuntos de estrellas que emparejen con el patrón de consulta es unir directamente las *buckets* de elementos candidatos basándose en en las distancias a pares correspondientes entre los elementos del patrón de consulta. Para construir una solución candidata, se ve a cada *bucket* como una relación, teniendo como esquema las coordenadas espaciales y el *id* de cada estrella, $B_i(starid, rad, dec)$. Una solución se obtiene siempre que se produzca una tupla que tenga un elemento vecino de cada *bucket*, tal que las distancias entre cada elemento en la solución hagan *distance-match*, es decir, emparejen a aquellas entre los elementos respectivos de la consulta, dentro de un radio de error ϵ. *Bucket_SJ* realiza un algoritmo *spatial nested loop* para recorrer las *buckets* de candidatos y comprueba predicados de distancia. Específicamente, la restricción *distance-match* corresponde a aplicar un *cyclic join* entre todas las *buckets* del conjunto de *buckets* seguido por un filtrado entre no vecinos en el ciclo. Por ejemplo, *Bucket_SJ* encontraría a las parejas (t_n, t_{n+1}) con $t_n \in B_n$ y $t_{n+1} \in B_{n+1}$ si $\left\|q_i - q_{i+1}\right\| - \epsilon \leq \|t_n - t_{n+1}\| \leq \left\|q_i - q_{i+1}\right\| + \epsilon$. Entonces, dados esos pares $(B_1, B_2), \ldots, (B_k, B_1)$, *Bucket_SJ* los unirá cíclicamente. Entonces, para cada k-tupla de estrellas $(s_1, s_2, \ldots, s_k)$ que sobrevivan al *join*, *Bucket_SJ* también comprobará las distancias de estrellas no vecinas (e.g., comprobar que $\left\|q_n - q_{n+3}\right\| - \epsilon \leq \|s_n - s_{n+3}\| \leq \left\|q_n - q_{n+3}\right\| + \epsilon$) .

2.3 CONSULTAS DE CONSTELACIÓN GENERALES

En esta sección se muestra cómo extender las consultas de constelación puras a consultas de constelación isotrópicas y no isotrópicas. Las consultas de constelación puras especifican a la vez las propiedades y las distancias de un patrón y requieren que cualquier secuencia de estrellas que empareje coincida con las distancias dentro de un radio de error ϵ. En esencia, las consultas de constelación puras encuentran colecciones de estrellas que emparejan a un patrón, permitiendo rotación y traslación.
Las *consultas de constelación isotrópicas* encuentran colecciones de estrellas que emparejan a un patrón, permitiendo rotación, traslación, y escalado lineal (isotrópico). Ignorando el error por un momento, un patrón cuadrado escalaría a un cuadrado de grandes dimensiones en el firmamento. Formalmente, las consultas de constelación isotrópicas permiten que $(s_1, s_2, \ldots, s_k)$ empareje a un patrón $(q_1, q_2, \ldots, q_k)$ si existe algún factor de escala f tal que para toda $1 \leq i, j \leq k$,
$\left\|q_i - q_k\right\| - \epsilon \leq \left\|s_i - s_j\right\| \leq \left\|q_i - q_k\right\| + \epsilon$
Las *consultas de constelación no isotrópicas* encuentran colecciones de estrellas que emparejan a un patrón, permitiendo rotación, traslación, y escalamiento alabeado. Esto es, algunas distancias podrían ser escaladas de manera distinta que otras. Por ejemplo, nuevamente ignorando el error, un patrón cuadrado en el cual un lado podría expandirse o contraerse podría emparejar a un trapezoide.

2.3.1 *Consultas de Constelación Isotrópicas*

Dado que en las consultas de constelación isotrópicas el factor de escalado f puede tomar cualquier valor real, el reto en el método de [7] es encontrar un conjunto suficientemente discreto de factores de escala que imitará el

intentar encontrar la infinidad no numerable de posibles valores de f de manera eficiente.
Se ilustran con un ejemplo las dificultades, y se apunta en la dirección de una solución. Supóngase que se busca un triángulo equilátero consistente de los puntos p_1, p_2, p_3. Como este es un triángulo equilátero, todas las distancias intra-patrón son la misma. Entonces, se puede hacer 1 la distancia máxima intra-patrón sin pérdida de generalidad. Supóngase que el término de error aditivo tiene como valor $\epsilon = 2$. Si las estrellas s_1, s_2, s_3 tienen las siguientes distancias a pares $\|s_1 - s_2\| = 8, \|s_2 - s_3\| = 12, \|s_3 - s_1\| = 12$, entonces el algoritmo podría emparejar q_1 y q_2 a s_1 y s_2. Como consecuencia, un algoritmo ingenuo podría intentar poner como factor de escalado $\|s_1 - s_2\| / \|q_1 - q_2\| = 8$. Sin embargo, un factor $f = 8$ no funcionaría dado los lados de longitud 12. En su lugar, se quiere un factor de escalado $f = 10$, aunque ningunas dos estrellas están a distancia 10 en el patrón de consulta. Se discute cómo sucede esto a continuación.
El método de [7] comienza escogiendo dos estrellas s_1 y s_2 y asignándole el valor $\|s_1 - s_2\| / \|q_1 - q_2\|$ a un factor de escala candidato llamado *scalebasic*. Las siguiente condiciones expresan restricciones sobre cualquier factor de escala final f que mapee s_1 y s_2 a q_1 y q_2. (Dado que las distancias en el patrón son relativas, en lo subsecuente, se considera $\|q_1 - q_2\| = 1$, sin pérdida de generalidad. Esto simplifica la notación):

CONDICIÓN f ISOTRÓPICA: Si $\|q_1 - q_2\| = 1$ y $scalebasic = \|s_1 - s_2\| / \|q_1 - q_2\|$, entonces f estará entre los valores $f_{min} = scalebasic - \epsilon$ y $f_{max} = scalebasic + \epsilon$.

LONGITUDES ISOTRÓPICAS ACEPTABLES: Para puntos del patrón q_i, q_j, la distancia aceptable para una pareja de estrellas correspondiente s_i y s_j debe estar entre $f_{min} \times \|p_i - p_j\| - \epsilon$ y $f_{max} \times \|p_i - p_j\| + \epsilon$.

Estas dos condiciones permitirán encontrar a todas las colecciones de estrellas que emparejen con la colección patrón dentro de un factor de escala f y un error aditivo ϵ y permitirán especificar a f.

2.3.1.1 *Algoritmo de Consulta de Constelación Isotrópica*

El algoritmo consta de seis pasos, comenzando con un patrón de k ubicaciones y una cota de error aditivo ϵ. La consulta podría tener algunas otras restricciones, como un factor de escala posible mínimo o máximo. Otras restricciones podrían involucrar atributos no espaciales. Se les llamará C colectivamente a esas restricciones:

(i) encontrar a la pareja de puntos más distantes en la consulta, a los que se denotará como q_1 y q_2, y se establece su distancia, sin pérdida de generalidad, como 1.

(ii) Para cada pareja de estrellas que satisfaga las restricciones C, llámense s_1 y s_2, se establece un factor de escala *scalebasic*: $sb = \|s_1 - s_2\| / \|q_1 - q_2\|$. Luego s_1 y s_2 serían estrellas candidatas a emparejar a q_1 y q_2. Se calcula f_{min} y f_{max} de acuerdo a la **condición f isotrópica**.

(iii) Se incluye a una estrella candidata s como posible coincidencia del punto del patrón de consulta q_j (para $j \neq 1, 2$) si $\|s_1 - s\|$ y $\|s_2 - s\|$ se ajustan a la **condición de longitudes isotrópicas aceptables** con

respecto a $\left\|q_1 - q_j\right\|$ y $\left\|q_2 - q_j\right\|$. Todas las estrellas tales irán a la *bucket* B_j.

(iv) La *bucket* B_1 consta únicamente de s_1, y la *bucket* B_2 consta únicamente de s_2. Todas las demás casillas se calculan como se indicó en el paso anterior.

(v) Se realiza una *k-1 spatial join* como en el algoritmo de consulta de constelación pura.

(vi) Post-proceso. Cualquier secuencia de estrellas coincidentes debe ser validada de acuerdo a la cota de error ϵ. Para cada i, j tales que $1 \leq i, j \leq k$ se determinan los factores de escala mínimos y máximos $minscale_{i,j}$ y $maxscale_{i,j}$ tales que

$$\left\|s_i - s_j\right\| = (minscale_{i,j} \times \left\|q_i - q_j\right\|) + \epsilon$$

$$\left\|s_i - s_j\right\| = (maxscale_{i,j} \times \left\|q_i - q_j\right\|) - \epsilon$$

Se denota a la máxima de las escalas mínimas *MaxMin* y a la mínima de las escalas máximas *MinMax*. Si $MaxMin \leq MinMax$, entonces cualquier valor v en el rango entre *MaxMin* y *MinMax* será un factor de escala satisfactorio siempre que satisfaga la **condición f isotrópica**, esto es, que el valor v esté dentro de una distancia ϵ de *scalebasic*. De otra forma no existe un factor de escala satisfactorio y el candidato a solución es un falso positivo, así que es descartado.

2.4 CONSULTAS DE CONSTELACIÓN NO ISOTRÓPICAS

Recordemos que las consultas de constelación puras encuentran constelaciones de estrellas que coinciden con un patrón, permitiendo rotación y traslación. Las consultas de constelación isotrópicas encuentran colecciones de estrellas que coinciden con un patrón, permitiendo traslación, rotación, y escalamiento linear (isotrópico). Las consultas de constelación no isotrópicas encuentran colecciones de estrellas que coinciden con un patrón, permitiendo rotación, traslación, y escalado no linear (alabeado).
Formalmente, las consultas de constelación generales no isotrópicas permiten que $(s_1, s_2, \ldots, s_k)$ coincida con un patrón $(q_1, q_2, \ldots, q_k)$ si existe un factor de escala f y algún conjunto de modificadores del factor de escala .^estiradores"$m_{i,j} \geq 1$ tales que
$f \times \left\|q_i - q_j\right\| - \epsilon \leq \left\|s_i - s_j\right\| \leq f \times m_{i,j} \times \left\|q_i - q_j\right\| + \epsilon$.
En contraste con el caso isotrópico, el factor de escala para parejas distintas de puntos del patrón podría diferir. Esto es, el caso isotrópico es un caso especial en el cual todos los $m_{i,j} = 1$. El reto, de nuevo, es encontrar a un conjunto lo suficientemente discreto que imitará el probar la infinidad de posibles factores de escala y de valores de estiramiento (e.g., entre 1 y $m_{i,j}$) de manera eficiente.
Las fórmulas para este caso generalizan directamente al caso isotrópico (para el cual $m_{i,j} = 1$). De nuevo, se comienza por escoger dos estrellas s_1 y s_2 y se establece *scalebasic* como $\|s_1 - s_2\| / \left\|q_1 - q_2\right\|$. Supóngase de nuevo sin pérdida de generalidad que $\left\|q_1 - q_2\right\| = 1$.

CONDICIÓN f NO-ISOTRÓPICA: f se encuentra entre un valor mínimo $f_{min} = (scalebasic - \epsilon)/m_{1,2}$ y un valor máximo $f_{max} = scalebasic + \epsilon$.

LONGITUDES ACEPTABLES NO-ISOTRÓPICAS La longitud aceptable para algún otro lado s_i, s_j se encuentra entre $(f_{min} \times \left\|q_i - q_j\right\|) - \epsilon$ y $(f_{max} \times m_{i,j} \times \left\|q_i - q_j\right\|) + \epsilon$.

2.4.1 *Algoritmo de consulta de constelación no-isotrópica*

De nuevo, se usa un algoritmo de seis pasos, dada una cota de error aditivo ϵ junto con un conjunto de factores de estiramiento $m_{i,j}$.

(i) Se toma como pareja base p_m y p_n si $\|p_m - p_n\|$ es máxima sobre todas las parejas p_i, p_j. Sin pérdida de generalidad, se designa a $m = 1$ y a $n = 2$ por consistencia con el caso isotrópico anterior. Se designa $\left\|q_1 - q_2\right\| = 1$. Se calculan f_{min} y f_{max} de acuerdo a la **condición** f **no-isotrópica**.

(ii) Para cada pareja de estrellas s_1 y s_2 que satisfagan al conjunto de restricciones C, se designa al factor de escala $scalebasic = \|s_1 - s_2\| / \left\|q_1 - q_2\right\|$. Esto es exactamente como en el caso isotrópico.

(iii) Para que una estrella candidata s corresponda a un punto q_j del patrón (para $j \neq 1, 2$), $\|s_1 - s\|$ y $\|s_2 - s\|$ deben cumplir con la condición de **longitudes aceptables no-isotrópicas** respecto a $\left\|q_1 - q_j\right\|$ y $\left\|q_2 - q_j\right\|$. Las estrellas tales entran en la *bucket* B_j.

(iv) (como en el caso isotrópico) la *bucket* B_1 consiste únicamente de s_1, y B_2 consiste únicamente de s_2. Todas las demás casillas podrían consistir de cero o más estrellas basándose en cada punto q_j del patrón de consulta como se calcula en el paso anterior.

(v) (como en el caso isotrópico) Realizar un *bucketed spatial join* como en el caso del algoritmo de consulta de constelación pura.

(vi) Post-procesamiento: Cualquier secuencia de estrellas coincidentes debe ser validada con respecto a una cota de error ϵ y a los factores de estiramiento $m_{i,j}$. Esto significa que debe existir una f tal que para todo lado q_i, q_j, $f \times \left\|q_i - q_j\right\| - \epsilon \leq \left\|s_i - s_j\right\| \leq f \times m_{i,j} \times \left\|q_i - q_j\right\| + \epsilon$. Inspirado por el paso de post-proceso en el caso isotrópico, para cada pareja s_i, s_j se encuentra el menor factor de escala posible, y luego el mayor factor de escala posible. El menor factor de escala posible para una pareja i, j es $g_{i,j}(min) = min_g\{(g \times m_{i,j} \times \left\|q_i - q_j\right\| + \epsilon \geq \left\|s_i - s_j\right\|)\}$. Luego $g_{i,j}(min) = (\left\|s_i - s_j\right\| - \epsilon) / (m_{i,j} \times \left\|q_i - q_j\right\|)$. Intuitivamente, el menor factor de escala posible es aquel que estira de p_i a p_j tanto como sea posible y usa la cota de error aditivo ϵ para hacer que $\left\|q_i - q_j\right\|$ alcance a $\left\|s_i - s_j\right\|$. Recíprocamente (pero sin el factor de estirado), $g_{i,j}(max) = max_g\{(g \times \left\|q_i - q_j\right\|) - \epsilon \leq \left\|s_i - s_j\right\|\}$. Así que $g_{i,j}(max) = (\left\|s_i - s_j\right\| + \epsilon) / \left\|q_i - q_j\right\|$.
Procediendo como en el caso isotrópico, para todas i, j, se denota como *MaxMin* a la máxima de las $g_{i,j}(min)$ y *MinMax* a la mínima de

las $g_{i,j}(max)$. Si $MaxMin \leq MinMax$ entonces cualquier valor v en el rango entre $MaxMin$ y $MinMax$ será un factor de escala satisfactorio provisto que satisfaga la **condición f no-isotrópica**. esto es, siempre que se encuentre entre $(scalebasic - \epsilon)/m_{i,j}$ y $scalebasic + \epsilon$. De otra manera no existe factor de escala satisfactorio.

3

INVARIANTES AFINES DE POLÍGONOS GENERALIZADOS Y EMPAREJAMIENTO BAJO TRANSFORMACIONES AFINES

3.1 INTRODUCCIÓN

En este capítulo se presentan las invariantes afines de polígonos generalizados y sus aplicaciones para emparejamientos de estos de [3].

3.1.1 *Invariantes*

En lo que sucesivo se fija una $n \geq 3$.

3.1.1.1 *El problema: emparejamiento indexado de polígonos*

Se identificará a puntos $(x, y) \in \mathbb{R}^2$ con números complejos correspondientes $z = x + iy \in \mathbb{C}$, donde i es la unidad imaginaria, la cual satisface la relación $i^2 = -1$. Un *polígono (generalizado)* en el plano se representa como un conjunto ordenado de puntos, o números complejos, donde el *orden especifica vértices consecutivos. Las auto intersecciones son permitidas*. Nótese que son etiquetas distintas para el polígono de n lados $(z_1, z_2, \ldots, z_{n-1}, z_n)$ los desplazamientos cíclicos $(z_2, z_3, \ldots, z_n, z_1), \ldots, (z_3, z_4, \ldots, z_1, z_2), \ldots, (z_n, z_1, \ldots, z_{n-2}, z_{n-1})$, dependiendo del vértice en el que se empiezan a listar estos. El mismo polígono también podría ser listado en orden inverso, a saber: $(z_n, z_{n-1}, \ldots, z_2, z_1), \ldots, (z_1, z_n, \ldots, z_3, z_2), \ldots, (z_{n-1}, z_{n-2}, \ldots, z_1, z_n)$. Se le llamará *reetiquetado cíclico* a cualquiera de aquellas etiquetas distintas del mismo polígono.
Una transformación afín $f : \mathbb{R}^2 \to \mathbb{R}^2$ es una composición de un isomorfismo lineal y una traslación, esto es

$$f\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} + \begin{pmatrix} r \\ s \end{pmatrix}$$

donde $ad - bc \neq 0$. Haciendo

$$\alpha = \frac{a+d}{2} + i\frac{c-b}{2}, \beta = \frac{a-d}{2} + i\frac{c+b}{2}, \gamma = r + is,$$

la transformación afín f puede ser escrita en términos de sumas y productos de números complejos de la forma

$$f(z) = \alpha z + \beta \overline{z} + \gamma$$

donde $\overline{z}$ es el conjugado complejo de z.

Definición 2. *Para cada $j \in \mathbb{Z}$ se considera la función $\varphi_j : \mathbb{C}^n \to \mathbb{C} \cup \{\infty\}$ dada por*

$$\varphi_j : (z_1, \ldots, s_n) = \frac{\sum_{k=1}^{n} \lambda^{jk} z_k}{\sum_{k=1}^{n} \lambda^{-jk} z_k}$$

donde $\lambda = e^{2\pi i/n}$ es una n-ésima raíz de la unidad.

Proposición 1. *φ_j es invariante bajo la acción de transformaciones de semejanza que preserven la orientación en polígonos con n vértices: esto es, si $\alpha, \gamma \in \mathbb{C}$ con $\alpha \neq 0$, entonces*

$$\varphi_j(\alpha z_1 + \gamma, \alpha z_2 + \gamma, \ldots, \alpha z_n + \gamma) = \phi_j(z_1, z_2, \ldots, z_n).$$

Observación 1. *Las funciones φ_{jn}, con $j \in \mathbb{Z}$, son las únicas funciones constantes cuando n es un entero impar, y las funciones $\varphi_{\frac{jn}{2}}$, con $j \in \mathbb{Z}$ son las únicas funciones constantes para n par. En otras palabras, φ_j es una función no constante exactamente cuando $j \in \mathbb{Z} \setminus \frac{n}{2}\mathbb{Z}$. Además, se puede decir que existen esencialmente tan solo $\lfloor (n-1)/2 \rfloor$ funciones φ_j distintas, ya que*

$$\varphi_{j+n} = \varphi_j \text{ y } \varphi_{n-j}\varphi_j = 1, \text{ para toda } j \in \mathbb{Z}.$$

De hecho, $\varphi_1, \varphi_2, \ldots, \varphi\lfloor (n-1)/2 \rfloor$ es un conjunto representativo de todas las funciones.

Observación 2. *Cada $Z = (z_1, \ldots, z_n) \in \mathbb{C}^n$ puede ser expresado como una combinación lineal $Z = \sum_{k=1}^{n} x_k E_k$ donde*

$$x_k = \frac{1}{n} \sum_{l=1}^{n} \lambda^{-kl} z_l$$

y

$$E_k = (\lambda^k, \lambda^{2k}, \ldots, \lambda^{nk}), k = 1, \ldots, n.$$

De hecho, $\{E_1, \ldots, E_n\}$ forma una base para el espacio vectorial complejo $\mathbb{C}^n$. Visto geométricamente, cada E_k es un polígono generalizado regular inscrito en el círculo unitario. Si $Z \in \mathbb{C}^n \setminus \{0\}$, entonces existe un entero j tal que $x_j \neq 0$ dado que $\{E_1, \ldots, E_n\}$ es una base, por lo tanto $\sum_{l=1}^{n} \lambda^{-jl} z_l \neq 0$ para tal j.

Observación 3. *Los conjuntos de nivel $\varphi^{-1}(c) = \{(z_1, \ldots, z_n) \in \mathbb{C}^n : \varphi_j(z_1, \ldots, z_n) = c\}$ son subvariedades complejas de dimensión $(n-1)$ con medida cero en $\mathbb{C}^n$, dado que cualquier punto en $\mathbb{C} \cup \{\infty\}$ es un valor regular de φ_j, para cada $j \in \mathbb{Z} \setminus \frac{n}{2}\mathbb{Z}$. Esto se sigue de un cálculo directo que muestra que $\frac{\partial \varphi_j}{\partial z_k} = 0 \implies \varphi_j = \lambda^{2jk}$. Entonces, dado cualquier polígono Z fijo la probabilidad de que un polígono aleatorio W satisfaga $\varphi_j(Z) = \varphi_j(W)$ es igual a cero.*

Proposición 2. *El comportamiento de φ_j bajo reetiquetado cíclico está dado por las fórmulas*

$$\varphi_j(z_2, z_3, \ldots, z_n, z_1) = \lambda^{-2j} \varphi_j(z_1, z_2, \ldots, z_n),$$
$$\varphi_j(z_n, z_{n-1}, \ldots, z_2, z_1) = \frac{\lambda^{2j}}{\varphi_j(z_1, z_2, \ldots, z_n)}$$

para toda $j \in \mathbb{Z} \setminus \frac{n}{2}\mathbb{Z}$. Si W es un reetiquetado cíclico de Z, la n-ésima potencia satisface las identidades

$$\varphi_j(Z)^n = \varphi_j(W)^n \text{ si las etiquetas tienen la misma orientación, y}$$
$$\varphi_j(Z)^n = \varphi_j(W)^{-n} \text{ si las etiquetas tienen orientaciones opuestas.}$$

3.2 DISTANCIA EUCLIDIANA ENTRE INVARIANTES DE PERTURBACIONES.

Dado un polígono $Z = (z_1, \ldots, z_n)$ y un número real $\rho > 0$, se considera una perturbación $Z + \Delta Z = (z_1 + \Delta z_1, \ldots, z_n + \Delta z_n)$ con $|\Delta z_k| < \rho$ para toda $k = 1, \ldots, n$. Ahora, se da una estimación para $|\varphi_j(z_1, \ldots, z_n) - \varphi_j(z_1 + \Delta z_1, \ldots, z_n + \Delta z_n)|$. Esta estimación proporciona una cota para la perturbación de la invariante en función de las perturbaciones en los vértices.

Proposición 3. *Sea $Z = (z_1, \ldots, z_n) \in \mathbb{C}^n \setminus \{0\}$. Considérese un entero $j \in \mathbb{Z} \setminus \frac{n}{2}\mathbb{Z}$ tal que $\sum_{l=1}^{n} \lambda^{-jl} z_l \neq 0$ (véase la observación 2). Sea ρ un número real positivo tal que $n\rho < \mu := |\sum_{l=1}^{n} \lambda^{-jl} z_l|$. Entonces para cada $\Delta Z = (\Delta z_1, \ldots, \Delta z_n)$ con $|\Delta z_k| < \rho$, $k = 1, \ldots, n$, se tiene*

$$|\varphi_j(z_1, \ldots, z_n) - \varphi_j(z_1 + \Delta z_1, \ldots, z_n + \Delta z_n)| \leq \frac{2n\rho \sum_{l=1}^{n} |z_l|}{\mu(\mu - n\rho)}$$

Observación 4. *En la notación de la observación 2 es posible encontrar una j para la cual $|\varphi_j(z_1, \ldots, z_n) - varphi_j(z_1 + \Delta z_1, \ldots, z_n + \Delta z_n)|$ sea mínima escogiendo un elemento maximal en el conjunto $\{|x_1|, |x_2|, \ldots, |x_{\lfloor (n-1)/2 \rfloor}|, |z_{\lfloor (n-1)/2 \rfloor + 1}|, \ldots, |x_{n-2}|, |x_{n-1}|\}$.*

3.3 ALGORITMOS

Los polígonos generalizados son conjuntos ordenados de vértices. Usando las funciones φ_j son mapeados a números complejos. El problema de emparejar bajo transformaciones de semejanza se convierte en uno de consultas geométricas. En lo subsecuente, se supondrá la existencia de algoritmos para resolver consultas de rango circular para puntos en el plano bajo la distancia Euclidiana. En [2] los autores presentaron un algoritmo determinista para preprocesar un conjunto de n puntos para resolver consultas de rango circular (entre otras aplicaciones) en tiempo $O(n \log n)$ y espacio $O(n)$. En [1] se dan algoritmos para reportar los k vecinos más cercanos de un punto de consulta o los k puntos en una consulta de rango circular en tiempo óptimo $O(k \log n)$ usando la estructura de datos de [2]. Será procesada y almacenada una colección de m polígonos generalizados de n lados $Z_1, Z_2, \ldots, Z_m$, y un polígono W será presentado al momento de la consulta.
En lo subsecuente, se usará $\varphi_j(Z) = \varphi_j(z_1, \ldots, z_n)$ y $\overline{W} = (\overline{w}_1, \ldots, \overline{w}_n)$, donde $\overline{w}$ representa al conjugado complejo de w.

3.3.1 *Emparejamiento exacto bajo semejanzas*

Para encontrar todos los Z_ℓ tales que $W = f(Z_\ell)$ para alguna transformación de semejanza desconocida f, en un paso de preproceso se calculan todas las parejas $(\ell, \varphi_j(z_\ell)^n)$ para $1 \leq \ell \leq m$ y $1 \leq j \leq \lfloor (n-1)/2 \rfloor$. Esto corresponde a todos los polígonos tales que $\varphi_j(Z_\ell)^n = \varphi(W)^n$ o $\varphi_j(Z_\ell)^n = \varphi(W)^{-n}$ (Proposiciones 1 y 2). Dado que la probabilidad de colisión es cero (Observación 3), todos los R polígonos mapeados a $\varphi_j(z_\ell)^n$ o $\varphi_j(z_\ell)^{-n}$ deberían ser semejantes al polígono de consulta W. Calcular $\varphi_j(W)$ puede ser realizado en $O(n)$ operaciones, y utilizando *hashing* es posible reportar los R polígonos semejantes a W en tiempo $O(R)$. Estrictamente es

necesario verificar W contra los R polígonos coincidentes en la colección. Sin embargo, dado que la probabilidad de colisión es cero, el algoritmo puede ser aleatorizado trivialmente evitando la verificación, y aún tendría probabilidad 1 de éxito suponiendo que los polígonos sigan requisitos apropiados de posición general. El tiempo de consulta total será entonces $O(nR)$ verificando y $O(n+R)$ en una versión aleatorizada del algoritmo. Nótese que la cota en el tiempo de ejecución del algoritmo es independiente de m, el tamaño de la colección.

3.3.2 *Emparejamiento bajo semejanzas en condiciones de ruido*

Una situación ligeramente más general es cuando existe una función de ruido desconocida que afecta al emparejamiento. La imagen del polígono de consulta es una transformación de semejanza más ruido, a saber $W = (f(z_1 + \Delta z_1), f(z_2 + \Delta z_2), \ldots, f(z_n + \Delta z_n))$. En este caso, en lugar de recuperar solamente los polígonos mapeados a $\varphi_j(W)$, se recuperan todos lo polígonos tales que $|\varphi_j(W) - \varphi_j(z_\ell)| \leq r$, usando la cota dada por la proposición 3.
Primero se calculan y almacenan las ternas $(\ell, j, \varphi_j(z_\ell))$ con $1 \leq \ell \leq m$ y j como en la observación 4, produciendo el radio de búsqueda menor. Se agrupan las ternas por índice j, cada grupo será consultado independientemente. Al momento de consulta se calcula $\varphi_j(W)$ para las $\lfloor (n-1)/2 \rfloor$ funciones φ_j distintas. Para cada grupo de ternas $(\ell, j, \varphi_j(z_\ell))$ se calcula la consulta de rango de búsqueda circular, centrada en $\varphi_j(W)$ y con radio

$$r = \frac{2n\rho \sum_{l=1}^{n} |z_l|}{|\sum_{l=1}^{n} \lambda^{-jl} z_l| (|\sum_{l=1}^{n} \lambda^{-jl} z_l| - n\rho)}$$

usando la cota dada por la proposición 3. Si el punto inicial es desconocido, entonces se deberían consultar todos los desplazamientos cíclicos de W. Sea $m = m_1 + m_2 + \cdots + m_s$, con m_j el número de ternas con la misma j, y $s = \lfloor (n-1)/2 \rfloor$. El tiempo total de consulta usando el algoritmo de consulta de rango circular de [1] será
$k_1 \log(m_1) + k_2 \log(m_2) + \cdots + k_s \log(m_s) \leq sR \log(m)$, con k_i el número de polígonos en la consulta de rango circular para los grupos de ternas y R el resultado mayor en los grupos de ternas. Luego el tiempo total de consulta será $O(Rn^2 \log m)$ para el reetiquetado cíclico, y $O(m \log m)$ si la correspondencia es conocida.

4

CONSULTAS DE NUBES DE PUNTOS.

4.1 INTRODUCCIÓN

Se presentan dos métodos de consulta, uno para realizar consultas en un índice de nubes de puntos bidimensionales, y uno para realizar consultas de constelación dentro de un solo patrón de puntos bidimensionales.
La consultas usan un índice basado en la obtención de un polígono generalizado a partir de cada punto de cada patrón. Se obtienen las invariantes hiperbólicas de estos polígonos, y se almacena a cada una en una tupla junto con el número de vértices del polígono, el identificador del patrón del que proviene, y el identificador del punto dentro del patrón.

4.1.1 *Consultas sobre un índice de nubes de puntos.*

Dada una colección de nubes de puntos $P_1, \ldots, P_m$, donde $P_k = p_{k_1}, \ldots, p_{k_m}$, $p_{k_i} \in \mathbb{R}^2$ y un patrón de consulta $A = a_1, \ldots, a_m$, $a_k \in \mathbb{R}^2$ se busca alguna P_k tal que $f(P_k) = A$ para alguna transformación de semejanza f.

4.1.2 *Consultas de constelación dentro de un solo patrón.*

Dado un patrón de fondo $D = d_1, \ldots, d_m$, $d_i \in \mathbb{R}$, y un patrón de consulta $Q = q_1, \ldots, q_p$, $q_i \in \mathbb{R}^2$, se busca a un subconjunto $C \subseteq D$, y a una transformación de semejanza f tales que $f(Q) = C$.

4.2 DERIVANDO POLÍGONOS GENERALIZADOS DE LA TRIANGULACIÓN DE DELAUNAY.

Se describe un método para obtener polígonos generalizados invariantes bajo transformaciones de semejanza a partir de una nube de puntos $A = a_1, \ldots, a_m$, $a_i \in \mathbb{R}^2$. Estos polígonos hacen la vez de *tokens* de A para generar un índice invertido. De cada punto $a_i \in A$ se forma un polígono generalizado cuyos vértices son el mismo punto a_i, y sus vecinos en la triangulación de Delaunay de A de la siguiente manera:
Sea $a_i \in A$ y sean $a_{i_1}, \ldots, a_{i_{n-1}}$ sus vecinos en la triangulación de Delaunay, ordenados por distancia ascendente a a_i, esto es, $|a_i - a_{i_j}| < |a_i - a_{i_k}|$ para $1 \leq j < k \leq n-1$.

Definición 3. *Se le llama espiga de a_i al polígono*

$$E_i = (a_i, a_{i_1}, a_{i_2}, \ldots, a_{i_{n-1}}).$$

Ejemplo 1. *En la figura 4.1 se muestra el punto a_i, sus vecinos en la triangulación, y el polígono generalizado que es su espiga en la triangulación.*

Observación 5. *Si f es una transformación de semejanza, A es una nube de puntos en posición general y $a_i \in A$, entonces $espiga(f(a_i)) = f(espiga(a_i))$.*

Demostración. Sea $a_i \in A$. Dado que A está en posición general su triangulación de Delaunay es única. Al ser f una transformación de

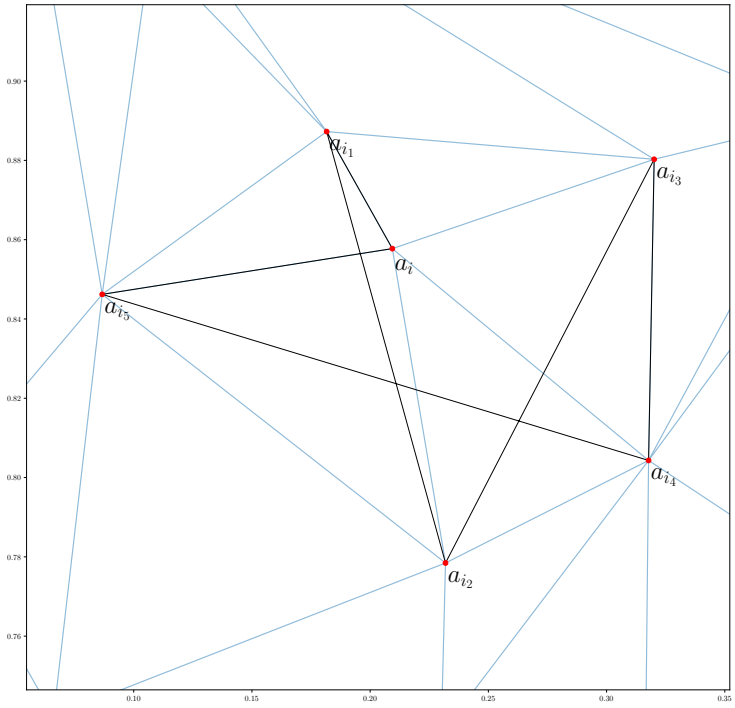


Figura 4.1: Espiga del punto a_i en negro, con la triangulación en azul.

semejanza preserva las proporciones entre distancias. Luego, si (a_i, a_j) es una arista en $\mathcal{DT}(A)$, entonces $(f(a_i), f(a_j))$ es una arista en $\mathcal{DT}(f(A))$. Y si $\left\|a_i - a_j\right\| \leq \|a_i - a_k\|$, entonces $\left\|f(a_i), f(a_j)\right\| \leq \left\|f(a_i) - f(a_k)\right\|$. Esto es, a_i conserva a todos sus vecinos después de la transformación de semejanza y las distancias entre a_i y sus vecinos mantiene el mismo orden. □

Entonces, las espigas pueden ser usadas a manera de *tokens* para indexar nubes bidimensionales de puntos bajo transformaciones de semejanza. El fin es obtener un conjunto de espigas a partir de una nube de puntos para calcular las invariantes de estas. Este conjunto de invariantes permiten indexar a la nube de puntos. Las comparaciones en el índice son por cercanía bajo la norma compleja.

Proposición 4. *φ_j es no determinada para polígonos regulares si $j \neq k$: Consideremos a*

$$E_k = (\lambda^k, \lambda^{2k}, \ldots, \lambda^{nk})$$

Por la observación 2 los coeficientes de Fourier correspondientes a E_k cumplen:

$$x_l = \begin{cases} 1, & \text{si } l = k \\ 0, & \text{de otra manera} \end{cases}$$

Dado que φ_j es un cociente de la forma x_{n-j}/x_j, $\varphi_j(E_k)$ está determinada sólo para $j = k$.
Como φ_j es invariante bajo transformaciones de semejanza, este resultado se extiende a todos lo polígonos regulares.

La discernibilidad de una invariante dependerá de la convexidad del polígono del cuál proviene. Los polígonos convexos son los menos discernibles. Les siguen los polígonos simples no-convexos, y finalmente los polígonos no-simples son los más discernibles. Estos últimos son los más distintos a los polígonos regulares en términos de sus coeficientes de Fourier. Nótese que el punto a_i podría no incluirse en la espiga E_i y esta seguiría funcionando como un *token* invariante bajo transformaciones de semejanza. Incluirlo aumenta la probabilidad de que el polígono E_i sea no-convexo y no-simple.

4.2.1 *Algoritmo*

Se presenta el algoritmo para obtener a la espiga correspondiente al k-ésimo punto de una nube de puntos.
El algoritmo recibe a $\mathcal{DT}(A)$, la triangulación de Delaunay de una nube de puntos A, y un número natural i, $0 \leq i < |A|$, y regresa un arreglo de puntos de A que forman a la espiga de a_k, su k-ésimo punto:

- Se obtiene a los vecinos de a_k en T, que se denominarán $N(k)$.
- Se crea un arreglo spk cuya primera entrada es a_k, y el resto de entradas es $N(k)$.
- Se regresa a spk ordenado por distancia a a_k.

4.3 ÍNDICES ESPACIALES DE INVARIANTES HIPERBÓLICAS

4.3.1 *Generación de Invariantes*

El algoritmo para generar las $\lfloor (n-1)/2 \rfloor$ invariantes no redundantes de un polígono generalizado de n vértices recibe un arreglo de números complejos P que representan al polígono, y regresa un arreglo de números complejos que contiene a las invariantes hiperbólicas.

- Se usa la FFT para generar $\{X_k\}$, la transformada discreta de Fourier de A, y $\{x_k\}$, la transformada discreta de Fourier inversa de A.
- Se regresa
$$\left\{ \frac{nx_k}{X_k} \right\}, k = 1, \ldots, \lfloor (n-1)/2 \rfloor$$

Dado que φ_j involucra divisiones entre coeficientes de Fourier de A, la efectividad de la implementación está sujeta a que la magnitud de estos sea lo suficientemente grande para no ocasionar problemas de estabilidad numérica. Además, aún en el caso de apareamiento sin ruido, debido al error que introduce la aritmética de punto flotante, es necesario considerar la búsqueda de la invariante dentro de un radio de tolerancia ϵ.

4.3.2 *Índice invertido de Nubes de Puntos*

Se presenta un índice invertido para realizar consultas de nubes de puntos bajo transformaciones de semejanza usando invariantes hiperbólicas.
Sea $A = a_1, \ldots, a_m$, $a_i \in \mathbb{R}^2$. Sea $\mathcal{DT}(A)$ la triangulación de Delaunay de A. Sea $e_i = \text{espiga}(\mathcal{DT}(A), i)$ y sea

$$\mathcal{E}(A) = \{e_i\}, i = 0, \ldots, |A| - 1,$$

esto es, $\mathcal{E}(A)$ es el conjunto de espigas de A. Se puede particionar a $\mathcal{E}(A)$ agrupando por el número n de vértices de cada e_i. Sea E un conjunto de espigas. Denotamos como $\mathcal{G}_n(E)$ al grupo de aquellas $e_i \in E$ que tienen n vértices. Sea $\mathcal{D}(E) = \{k \in \mathbb{N} : \mathcal{G}_k(E) \neq \emptyset\}$, esto es, el conjunto de los números de vértices cuyos grupos correspondientes son no-vacíos. Entonces, $\mathcal{D}(\mathcal{E}(A))$ es el conjunto de los distintos números de vértices que tienen las espigas de A.
A partir de una nube de puntos A se genera una colección de índices invertidos, uno por cada $n \in \mathcal{D}(\mathcal{E}(A))$. Las geometrías a indexar en cada uno de estos índices corresponden a las invariantes $u_{i,j} = \varphi_j(e_i)^n$, $j = 0, \ldots, \lfloor (n-1)/2 \rfloor - 1$, $e_i \in \mathcal{G}_n(\mathcal{E}(A))$. En el índice correspondiente a n, `V_n`, a cada espiga e_i de n vértices le corresponden $\lfloor (n-1)/2 \rfloor$ entradas, cada una con la siguiente estructura:

$$(\textit{geometry}, i, j, n)$$

- *geometry*, un punto con coordenadas $(\text{re}(u_{i,j}), \text{im}(u_{i,j}))$, o de haberse especificado un radio de error ϵ, un vecindario centrado en dicho punto con radio ϵ.
- `i`, el índice del punto a_i del que se generó la invariante.
- `j`, el exponente j en la función φ_j utilizada para generar a la invariante.
- `n`, el número de vértices de la espiga correspondiente a a_i.

La colección de índices se regresa como un diccionario en el que las llaves son las distintas `n` y los valores son los `V_n` correspondientes .

4.3.2.1 *Algoritmo*

El algoritmo recibe un arreglo de puntos bidimensionales A, y opcionalmente un radio de error ϵ, y regresa la colección de índices ya descrita.

- Se calcula $\mathcal{DT}(A)$, la triangulación de Delaunay de A.
- Para cada punto a_i de $\mathcal{DT}(A)$ se calculan las $u_{i,j}$, $0 \leq j < \lfloor (n-1)/2 \rfloor$.
- Se forma una tabla `U_i` con la siguiente estructura:
  - Una columna llamada `geometry` cuyas entradas son los puntos con coordenadas $(\text{re}(u_{i,j}), \text{im}(u_{i,j}))$, $j = 0, \ldots, \lfloor (n-1)/2 \rfloor$. O bien, si se provee un radio de tolerancia ϵ, cada punto de la columna `geometry` es expandido a un vecindario de radio ϵ centrado en dicho punto.
  - Una columna llamada `i` en la que las $\lfloor (n-1)/2 \rfloor$ entradas tienen todas por valor i.

  - Una columna llamada `n`, en la que las $\lfloor (n-1)/2 \rfloor$ entradas tienen por valor n.
  - Una columna `j`, en la que cada entrada tiene por valor el de la j de la φ_j correspondiente.
- Se concatena a las tablas `U_i` en una sola tabla `U`. Esta tabla se agrupa por la columna n, lo que resulta en tantas tablas `V_n` como números distintos de vértices tengan las espigas derivadas de A.
- Se regresa un diccionario en el que las llaves son las $n \in \mathcal{D}(\mathcal{E}(A))$, y los valores son las respectivas tablas `V_n`.

4.3.2.2 *Complejidad*

Denotamos como A a la nube de puntos a indexar, si $|A| = O(p)$, entonces el tiempo de construcción cada triangulación de Delaunay es $O(p \log p)$ [8]. El pre-procesamiento de una invariante consiste en ordenar los vecinos de un punto a por distancia a este, esto toma $O(\log n)$, donde n es el número de vecinos de a. De acuerdo a [10] el número de vecinos de un punto en una triangulación de un conjunto de puntos aleatorios está prácticamente acotado. Entonces este número se puede considerar como una constante pequeña. Luego, el tiempo que toma encontrar la espiga del punto y generar sus invariantes hiperbólicas es $O(1)$. Esto resulta en un GeoDataFrame con $O(p)$ invariantes.

4.3.3 *Índices de colecciones de nubes de puntos.*

Sea $\mathcal{P} = \{P_1, \ldots, P_m\}$ una colección de nubes de puntos bidimensionales finitas. A partir de $\mathcal{P}$ se genera, de manera semejante a la presentada en 4.3.2, una colección de índices de invariantes hiperbólicas. Recordemos que $\mathcal{E}(P_k)$ denota al conjunto de espigas derivadas de la nube de puntos P_k. Sea

$$\mathfrak{E} = \bigcup \{\mathcal{E}(P_k) : P_k \in \mathcal{P}\}$$

Esto es, $\mathfrak{E}$ es el conjunto de todas las espigas de las P_k. Ahora, cada índice contiene invariantes de las espigas de $\mathfrak{E}$, y hay tantos índices como números de vértices distintos tengan estas espigas. Es decir, si

$$\mathfrak{D} = \mathcal{D}(\mathfrak{E}),$$

entonces hay un índice por cada $n \in \mathfrak{D}$. En el índice `V_n` correspondiente a cada n, por cada $u_{i,j,k} = \varphi_j(\text{espiga}(\mathcal{DT}(P_k), i))^n$, $j = 0, \ldots, \lfloor (n-1)/2 \rfloor - 1$ hay un registro con la geometría que representa a la invariante hiperbólica, y los valores de i, j , n y k.
El algoritmo regresa un diccionario en el que las llaves son las $n \in \mathfrak{D}$ y los valores son los respectivos índices.
Cada índice que se regresa tiene la siguiente estructura:

$$(geometry, i, j, n, k)$$

Donde

- `geometry` es un punto con coordenadas $(\text{re}(u_{i,j,k}), \text{im}(u_{i,j,k}))$. O bien, si se provee un radio de error ϵ, el punto es expandido a un vecindario de radio ϵ centrado en dicho punto .

- `i` es el valor de i, el identificador de la espiga dentro de P_k.
- `j` es el valor del exponente j en la función φ_j usada para generar a la invariante.
- `n` es el valor de n, el número de vértice de la espiga.
- `k` es el valor es k, el índice de P_k.

4.3.3.1 *Algoritmo*

El algoritmo recibe una colección de arreglos de puntos bidimensionales finitos $\mathcal{P} = \{P_1, \ldots, P_m\}$ y regresa la colección de índices arriba descrita como un diccionario.

- De cada $P_k \in \mathcal{P}$ se genera un índice `U_k` como el de 4.3.2. A `U_k` se le agrega una una columna `k`, cuyo valor en todos los registros es k.
- Se concantena a las tablas `U_k` para formar una tabla `U`.
- `U` se agrupa por la columna `n` para formar las tablas `V_n` que serán los índices correspondientes a los número de vértices distintos que tiene las espigas.
- Se regresa un diccionario en el que las llaves son las $n \in \mathfrak{D}$, y los valores son los respectivos `V_n`.

4.3.3.2 *Complejidad*

El índice espacial que se crea sobre $O(mp)$ invariantes toma $O(mp \log mp)$ usando STR Bulk Loading [6]. Entonces, el tiempo de construcción total del índice de invariantes a partir de un conjunto de puntos es $O(mp \log mp)$.

4.3.4 *Consultas de constelación*

Se busca responder la pregunta de si, dados un patrón de fondo finito y bidimensional D, y un patrón de consulta, también finito y bidimensional Q, existe un función de semejanza f tal que $f(Q) \subseteq D$. Para esto se genera una colección de índices `D_idx` para D con el procedimiento de 4.3 con $\epsilon = 0$, es decir, las invariantes se representan como puntos. Las invariantes del patrón de consulta Q serán representadas no como puntos, sino como vecindarios de algún radio de tolerancia ϵ. Esto permite definir el radio de tolerancia al momento de la consulta.

4.3.4.1 *Método de consulta.*

Se tiene previamente generado un índice espacial para el patrón de fondo D usando el método de 4.3.2. Se recibe un patrón de consulta Q.

1. Se genera un índice `Q_idx` a partir de Q con $\epsilon > 0$ para manejar el ruido. Esto resulta en un índice en el que las invariantes son vecindarios de radio ϵ centrados en cada invariante $u_{i,j}$.
2. Para cada pareja llave-valor `(n, V_n)` de `Q_idx`, con `n` $\in \mathcal{D}(\mathcal{E}(Q))$, y `V_n` la tabla correspondiente a las invariantes de las espigas con `n` vértices, se verifica si en `D_idx` existe un también un índice `W_n` con llave `n`. Si lo hay, se realiza un *spatial join* de tipo *inner* con *intersects*

como operador espacial entre `V_n` como tabla izquierda y `W_n` como tabla derecha, que regresa una tabla `candidatos_n` con los puntos de `W_n` que se encuentren contenidos en algún vecindario de `V_n`.

3. Se concatena a las tablas `candidatos_n` en una tabla `candidatos`. `candidatos` tiene como estructura:

   (`index_left`, `i_left`, `j_left`, `n_left`, `geometry`,
   `index_right`, `i_right`, `j_right`, `n_right`)

   donde:

   - `index_left`. Es el índice de la invariante en `V_n`.
   - `i_left`. El identificador del punto en D del que se generó la espiga de `V_n` que fue emparejada.
   - `j_left`. El exponente j de la función $\varphi_j(espiga)^n$ que generó a la invariante en `V_n`.
   - `n_left`. Es el número de vértices de la espiga de `V_n`.
   - `geometry`. Es el punto que representa a la invariante en `V_n`.
   - `index_right`. Es el índice en `W_n` de la invariante emparejada.
   - `i_right` Es el identificador del punto en Q del que se generó la espiga de `W_n` que fue emparejada.
   - `j_right` Es el exponente j de la función $\varphi_j(espiga)^n$ que generó a la invariante en `W_n`.
   - `n_right` Es el número de vértices de la espiga de `W_n`.

4. Para obtener los emparejamientos potenciales entre los puntos de Q y D se agrupa la tabla `candidatos` por la pareja de columnas `['polygon_id_left','polygon_id_right']` y se cuenta el tamaño de cada grupo. Este conteo permite discriminar entre emparejamientos para una misma espiga, el más probable es el que tenga un conteo mayor. Luego, a partir de los emparejamientos de espigas se forman los emparejamientos de puntos. Se forman arreglos de $4 \times \lfloor (n-1)/2 \rfloor$ para cada pareja de espigas apareadas, donde n es el número de vértices de las espigas. Las dos primeras columnas de este arreglo son las coordenadas (x, y) de los vértices de la espiga pertenecientes a Q, las dos últimas columnas son las coordenadas (x, y) vértices de la espiga perteneciente a D. Finalmente se concatenan estos arreglos, lo que resulta en un arreglo de 4 columnas, donde cada renglón representa a un apareamiento entre una pareja de puntos.

4.3.4.2 *Complejidad*

Si $|Q| = n$ y $|D| = m$, la creación del índice para Q lleva $O(n \log n)$ tiempo, y el índice generado tiene $O(n)$ invariantes. Para los *spatial joins* entre las `V_n` y los `W_n`, toma $O(n \log m)$. La implementación del spatial join de GeoPandas crea el índice espacial del GeoDataFrame con mayor número de invariantes, en este caso `V_n`, y realiza una búsqueda de tipo *intersects* de cada invariante de `V_n` en el `W_n` correspondiente [4].

4.3.5 *Consultas en un Índice de Colecciones de Nubes de Puntos*

Dada una nube de puntos A y un conjunto de nubes de puntos $\{P_1, \ldots, P_m\}$ se describe un método para responder la consulta de si existen una transformación de semejanza f y una nube de puntos P_k tales que $A = f(P_k)$
Se usa el método de 4.3.2 para generar una colección de índices de invariantes de A. Esto resulta en un diccionario de índices `A_idx` en el que las llaves son los distintos número de vértices de las espigas de A, y los valores son los índices invertidos correspondientes. Y se usa el método de 4.3.3 con $\{P_1, \ldots, P_m\}$ como entrada para generar el diccionario de índices `P_idx`.
El método para realizar la consulta recibe a los dos diccionarios de índices, `A_idx`, que contiene a los índices de las invariantes correspondientes al patrón de consulta A, y `P_idx`, que contiene las invariantes de los patrones P_k a consultar, y se realiza una serie de *spatial joins* para obtener un candidato a emparejamiento:

- Se inicializa un diccionario vacío `candidatos`.
- Para cada pareja de llave-valor `(n, V_n)` del diccionario `A_idx` se prueba si existe la llave `n` en el diccionario `P_idx`. Si existe, se realiza un *spatial join* entre `V_n` y `P_idx[n]`. La tabla que resulta de este `spatial join` es `c`.
  - Si `c` es no-vacío, se hace `candidatos[n] = c`
- Si `candidatos` es no-vacío, se concatena a los `candidatos[n]` en una sola tabla `candidatos_df`.
- Se agrupa a `candidatos_df` por la columna k, el índice de las nubes de puntos `P_k`, y se obtiene k_{max}, el índice del grupo de mayor tamaño.

$P_{k_{max}}$ es entonces un candidato a emparejar a la nube de puntos A bajo alguna transformación de semejanza f.

4.3.6 *Complejidad*

Si $|A|$,$|P| = p$, y m es el número de nubes P_k en la colección , la creación del índice para A toma $O(p \log p)$, y contiene $O(p)$ invariantes. Cada `spatial join` toma $O(p \log pm)$. La implementación del spatial join de GeoPandas crea el índice espacial del GeoDataFrame con mayor número de invariantes, en este caso `idx_P[n]`, y realiza una consulta de tipo *intersects* entre cada invariante de `n_gdf` y `idx_P[n]` correspondiente [4].

4.3.7 *Limitaciones*

El que la espiga de un punto depende de los vecinos de este en la triangulación de Delaunay impone limitaciones sobre los patrones que se pueden emparejar con este método.
Supóngase que $f(Q) \subseteq D$ para alguna transformación de semejanza f. Para que el método pueda emparejar a Q con $f(Q)$, debe existir al menos un punto $q \in Q$ tal que si V_q es el conjunto de vecinos de q, entonces $f(V_q)$ es el conjunto de vecinos de $f(q)$ en S.

5 RESULTADOS

5.1 IMPLEMENTACIÓN

Las implementaciones de los algoritmos presentados en 4 fueron realizados en *Python*. La implementación de 4.3.1 utiliza la implementación de la *FFT* de *NumPy*. Para las implementaciones de 4.3.2 y 4.3.3 se utilizan *GeoDataFrames* de *Geopandas* como tablas con capacidad de realizar *spatial joins*.

5.2 BÚSQUEDA DE CONSTELACIÓN USANDO ÍNDICES DE INVARIANTES

Se prueba de manera empírica el desempeño del método de consulta de 4.3.4.
Se realizan experimentos variando los siguientes parámetros: el número de puntos del patrón de fondo, el número de puntos del patrón de consulta y el radio de tolerancia ϵ de las consultas de invariantes hiperbólicas. Para medir el desempeño se utilizan com métricas el tiempo de ejecución, la *precisión* y la *recuperación*. Se definen a continuación las dos últimas [5].

Definición 4. *La Precisión es la proporción de objetos recuperados que son relevantes*

$$Precisión = \frac{\#(Objetos\ relevantes\ recuperados)}{\#(Objetos\ recuperados)}$$

La recuperación se incrementa con el radio de tolerancia de la consulta de invariantes hiperbólicas, debido a la naturaleza de las operaciones de punto flotante.

Definición 5. *La Recuperación es la proporción de objetos recuperados relevantes*

$$Recuperación = \frac{\#(Objetos\ relevantes\ recuperados)}{\#(Objetos\ relevantes)}$$

5.2.1 *Configuración Experimental*

A grandes rasgos, la configuración experimental consta de los siguientes pasos:

- Se genera un conjunto D con m puntos escogidos aleatoriamente en la región $[0,1]^2$ que será usado como patrón de fondo. Este conjunto de puntos se almacena en un GeoDataFrame.
- Se genera la colección de índices de invariantes `D_idx` correspondientes a D a partir de su triangulación de Delaunay usando el método de 4.3.2.
- Para distintos valores de $k \geq 3$ y $\epsilon > 0$ se ejecutan l consultas:
  - El patrón de consulta se obtiene escogiendo un punto de manera aleatoria en $[0,1]^2$ y se toma a sus k vecinos más cercanos en D para formar una constelación de k puntos.

- A la constelación se le aplica una transformación de semejanza aleatoria f para formar al patrón de consulta Q.
- Se realiza una consulta con el método de 4.3.4 con Q como patrón de consulta, `D_idx` como índice de patrón de fondo y ϵ como radio de tolerancia, para intentar obtener parejas de puntos candidatas a ser un emparejamiento.

  De haber parejas candidatas, se estima una transformación de semejanza g.

- Se almacenan los siguientes parámetros y resultados de cada consulta:
  - `knn`: El número de puntos de la consulta.
  - `hausdorff`: La distancia de Hausdorff dirigida entre $g(P)$ y D, donde g es la transformación de semejanza candidata a ser f^{-1}.
  - `t`: El tiempo que tardó la consulta en nanosegundos.

  Si no hubo emparejamientos en la consulta el valor de `hausdorff` es `NaN`.

5.2.1.1 *Parámetros*

Se toman como valores de los parámetros $k = 3 \ldots, 20$, $l = 1000$, y $\epsilon = 10^{-16}, 10^{-8}, 10^{-4}, 10^{-3}$.
La transformación de semejanza aplicada rota por un ángulo aleatorio entre 0 y 2π, escala por un factor aleatorio entre 1 y 2, y finalmente traslada una distancia aleatoria entre 0 y 10 en las direcciones de x y y.
Se toma como un apareamiento exitoso aquel cuya distancia de Hausdorff sea menor a 10^{-14}

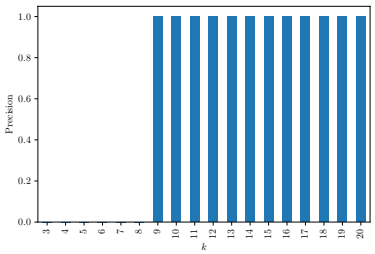
5.2.2 *Resultados*

En el conjunto de experimentos se varían los parámetros k, el número de puntos que conforman al patrón de consulta, y ϵ, el radio de búsqueda en el índice de invariantes. Intuitivamente, se espera que a mayor número de puntos en el patrón sea más probable conseguir un emparejamiento, dado que es más probable que un punto conserve a sus vecinos en la triangulación. También se espera que la tolerancia ϵ debería tener un valor lo suficientemente alto para obtener un buen desempeño en recuperación, pero no tanto como para conseguir falsos positivos.
En la figura 5.1 se muestra el desempeño en términos de precisión y recuperación para los distintos valores de ϵ, variando k.
Con $\epsilon = 10^{-16}$ en la figura 5.1a no hay emparejamientos para $k = 3, \ldots, 8$.
Para $k = 9, \ldots, 20$ la precisión es 1.0, pero la recuperación es minúscula (figura 5.1b), menor a 0.05 en todos los casos.
Con $\epsilon = 10^{-8}$ en la figura 5.1c sólo no hay emparejamientos para $k = 3$. La precisión se mantiene en 1.0 para aquellas k en las que sí ocurren emparejamientos. En la figura 5.1d la precisión sigue siendo minúscula para $k = 4$, y aumenta junto con k, llegando a ser muy cercana a 1.0 para $k = 12, \ldots, 17$, y se vuelve 1.0 para $k = 18, 19, 20$.
Con $\epsilon = 10^{-4}$ en la figura 5.1e y en la figura 5.1f la precisión y la recuperación se ven afectados disminuyendo conforme aumenta k.
En la figura 5.2 se muestra el desempeño en términos de tiempo, medido en nanosegundos.

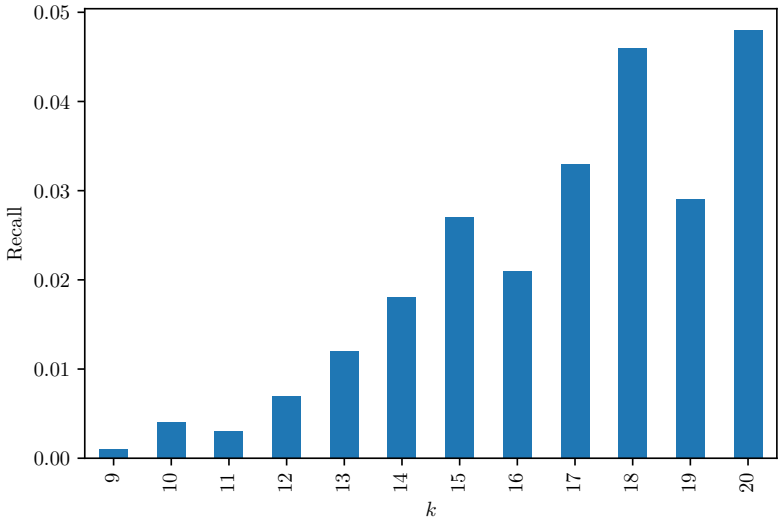
Con $\epsilon = 10^{-16}$ en la figura 5.2a la distribución de los tiempos de consulta se concentra alrededor del valor máximo. Esto corresponde a las consultas que no obtuvieron un candidato para el emparejamiento. Los valores de los tiempos de consulta que sí los obtuvieron aparecen como valore extremos. Con $\epsilon = 10^{-08}$ (figura 5.2b) cuando $k = 3$ al seguir sin haber candidatos para el emparejamiento los tiempos de consulta se mantienen todos alrededor del máximo. Con $k = 4, 5, 6$ la mayoría de los tiempos se mantienen en el máximo, correspondiendo con la mayoría de las consultas que no obtienen candidatos para el emparejamiento. Aparecen algunos puntos extremos alrededor de $t = 10^7$ns, correspondientes a los pocos emparejamientos conseguidos. Para $k = 7, 8, 9$ la distribución de los tiempos de consulta es sesgada, esto corresponde a la recuperación con valores alrededor de 0.5. En los casos $k = 7, 8$ la mayoría de los tiempos están alrededor del máximo, y suficientes están alrededor de 10^7ns como para segar la distribución. Con $k = 9$ sucede lo opuesto, la mayoría de los tiempos están alrededor de $t = 10^7$ns, correspondientes a la recuperación mayor a 0.5, y suficientes lo están alrededor de $t = 10^{18}$ns como para sesgar la distribución. A partir de $k = 10$ los tiempos se estabilizan alrededor de $t = 10^8$ns. En $k = 10, \ldots, 17$ todavía aparecen puntos extremos alrededor de $t = 10^{18}$ns. Con $k = 18, 19, 20$ ningún tiempo se aleja de $t = 10^8$ns, todas las consultas terminaron pronto arrojando candidatos para emparejamientos. Con $\epsilon = 10^{-04}$ el comportamiento en términos de tiempo de las consultas es muy parecido a aquel con $\epsilon = 10^{-8}$.

5.2.3 *Discusión*

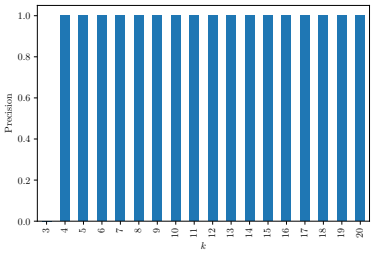
El método de emparejamiento tiene un desempeño bajo cuando el número de puntos en el patrón de consulta es menor a 12. Esto es consecuencia de la manera en la que se forman las espigas. Cuando $k = 3$ no se observó ningún emparejamiento, para que ocurra un emparejamiento con $k = 3$ los tres puntos tendrían que formar un triángulo en la envolvente convexa del patrón de fondo, cosa poco probable. Conforme aumenta el número de puntos en el patrón de consulta aumenta la probabilidad de que haya al menos una espiga en él que aparezca en el patrón de fondo después de la transformación de semejanza.
La precisión y la recuperación disminuyen cuando se pasa de $\epsilon = 10^{-8}$ a 10^{-4}. Era esperado que la precisión disminuyera, al ser más probable un falso positivo. La disminución de la recuperación es consecuencia directa de que disminuye también el número de verdaderos positivos, y dado que hubieron casos en los que la recuperación bajó de 1.0, patrones de consulta que eran encontrados con $\epsilon = 10^8$ dejaron de ser encontrados. Esto indica que la manera en la que se arman las parejas de espigas para recuperar la transformación puede mejorar.
Con todo, el desempeño es bueno para patrones de consulta a partir de un número de puntos lo bastante bajo para hacerlo aplicable. La precisión y la recuperación son altos, y dado que se usa un índice espacial, el desempeño en términos de tiempo es bueno conforme aumenta el número de puntos.



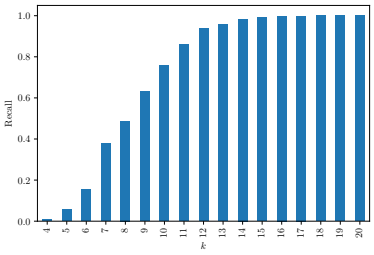
(a) Precisión $\epsilon = 1^{-16}$



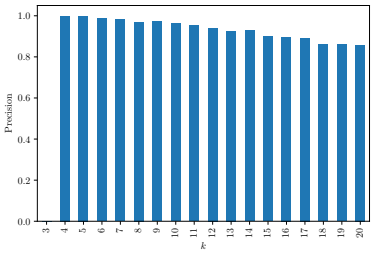
(b) Recuperación $\epsilon = 1^{-16}$



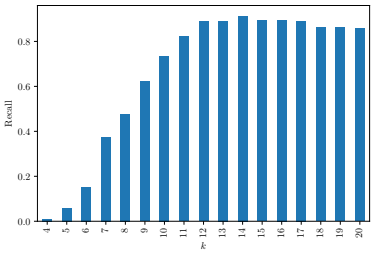
(c) Precisión $\epsilon = 1^{-8}$



(d) Recuperación $\epsilon = 1^{-8}$

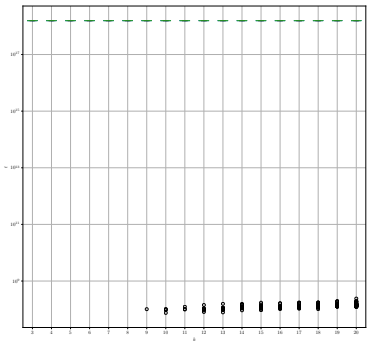


(e) Precisión $\epsilon = 1^{-4}$

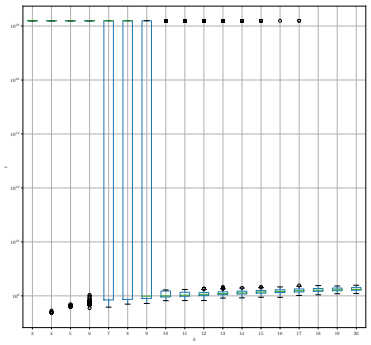


(f) Recuperación $\epsilon = 1^{-4}$

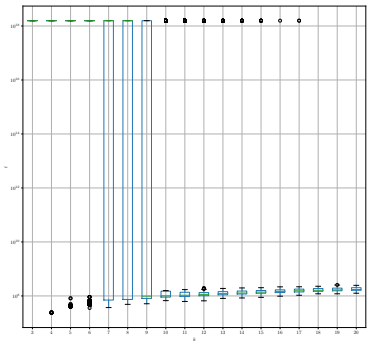
Figura 5.1: Precisión y Recuperación para distintos valores de ϵ.



(a) $\epsilon = 1^{-16}$



(b) $\epsilon = 1^{-8}$



(c) $\epsilon = 1^{-4}$

Figura 5.2: Tiempo de consulta para distintos valores de ϵ.

5.3 ÍNDICES DE NUBES DE PUNTOS.

Se prueba de manera empírica el desempeño de la construcción y consulta de índices de números de puntos, variando el tamaño de las nubes de puntos a indexar, y el número de nubes en el índice.

5.3.1 *Configuración Experimental*

A grandes rasgos, la configuración experimental consta de los siguientes pasos:

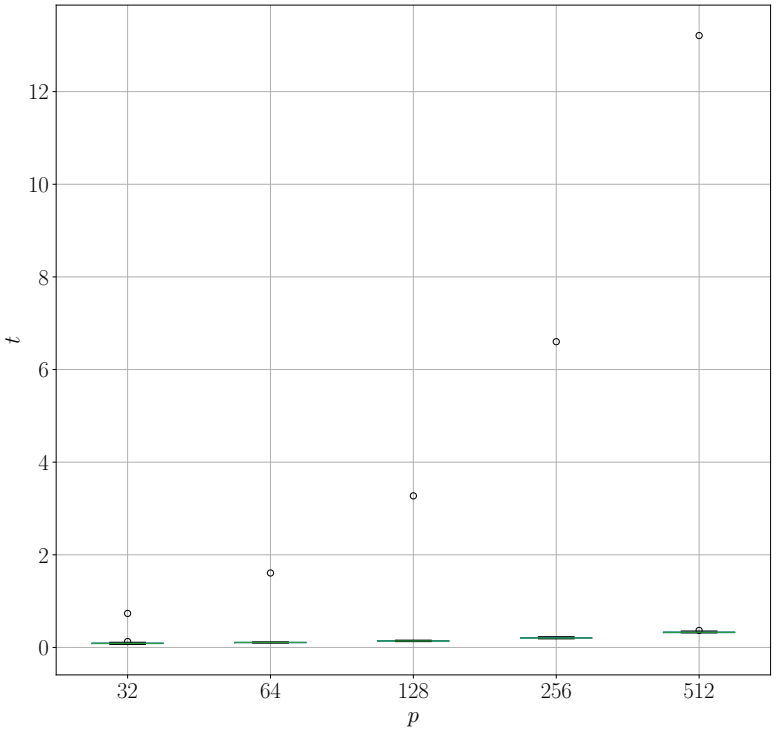
- Se generan m nubes de p puntos cada una. A esta colección de nubes de puntos se le genera una colección de índices con el método de la sección 4.3. Se registra el tiempo que toma la construcción de la colección.
- Se toma una muestra K de tamaño $\frac{m}{s}$ de los índices de las nubes de puntos. Para cada $k \in K$ se genera una transformación de semejanza aleatoria f_k. El factor de escala se toma aleatoriamente en $[1, 2]$. El ángulo de rotación de la transformación se toma aleatoriamente de $[0, 2\pi]$. El vector de translación se toma aleatoriamente de $[0, 10]^2$.
- Se calcula $A_k = f_k(B_k)$, y se realiza la consulta como se describe en ls subsección 4.3.5. Se registra el tiempo que toma la consulta, el índice k, y el índice candidato `match`.

5.3.1.1 *Parámetros*

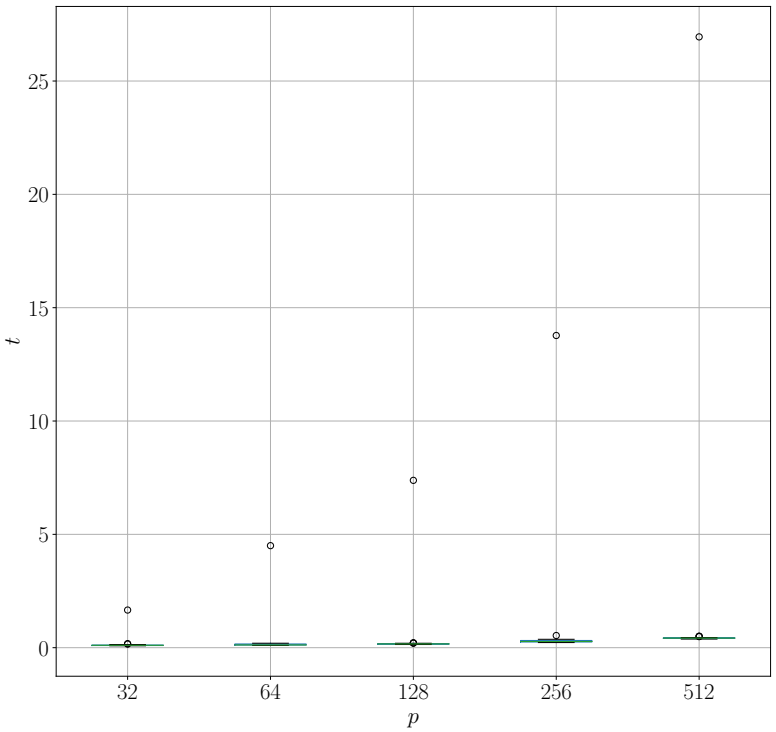
Se toman como valores de $m = 256, 512, 1024, 2048, 4096, 8192$, como valores de $p = 32, 64, 128, 256, 512$, y como valor de $s = 16$

5.3.1.2 *Resultados*

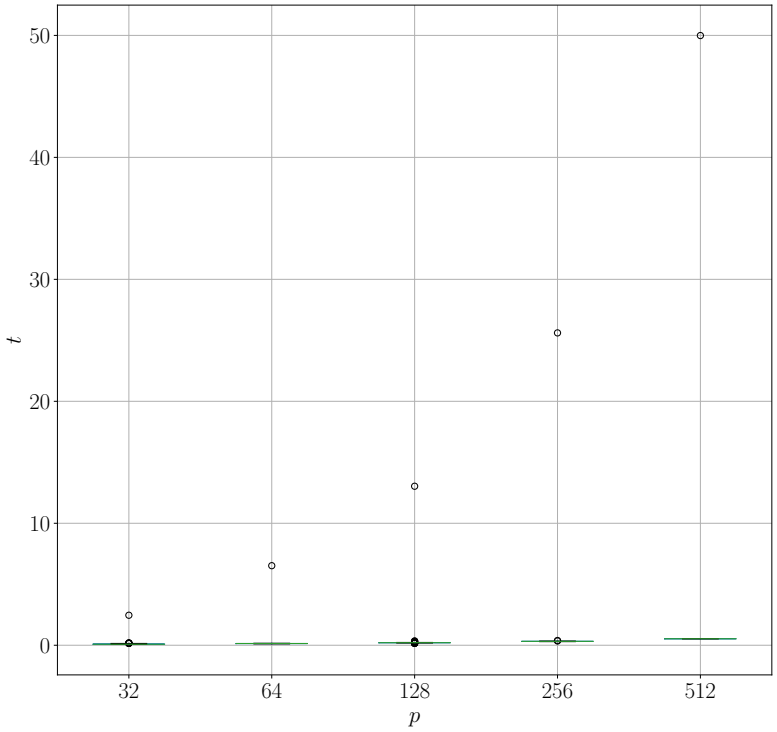
Los resultados de los tiempos de consulta se presentan de dos maneras distintas: dejando fijo el número de nubes de puntos m y variando el número p de puntos en cada nube; y dejando fijo el número de puntos p en cada nube y variando el número de nubes de puntos m.
En la figura 5.2 se muestran gráficas de caja para los valores fijos de $m = 256$ (subfigura 5.2d), $m = 512$ (subfigura 5.2e), $m = 1024$ (subfigura 5.2f), $m = 2048$ (subfigura 5.2g).
En la figura 5.3 se muestran gráficas de caja para los valores fijos de $p = 32$ (subfigura 5.3a), $p = 64$ (subfigura 5.3b), $p = 128$ (subfigura 5.3c), $p = 256$ (subfigura 5.3d), $p = 512$ (subfigura 5.3e)
El experimento para la pareja de parámetros $p = 512, m = 8192$ no concluyó por falta de recursos de hardware.
La primer consulta involucra la construcción del índice espacial R-Tree, el tiempo que toma esto se ve reflejado en los puntos extremos que aparecen en las gráficas. Cada una de las demás consultas toma menos de $1s$.
La precisión y el *recall* son 1 en todos los casos, en contraste con la búsqueda de sub-patrones. En la búsqueda de patrones completos todas las espigas de un patrón de consulta tienen una correspondencia en el índice.



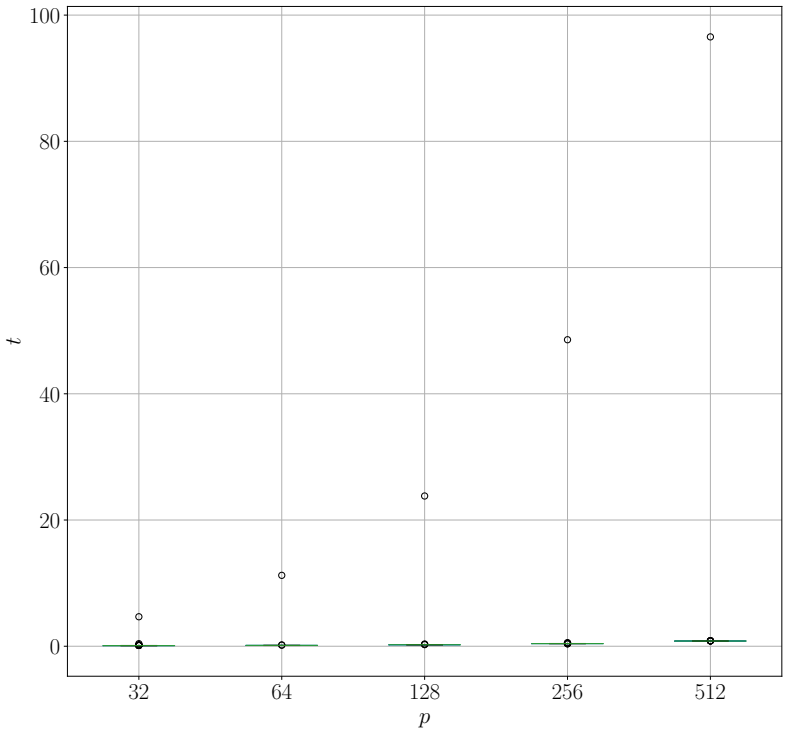
(d) $m = 256$



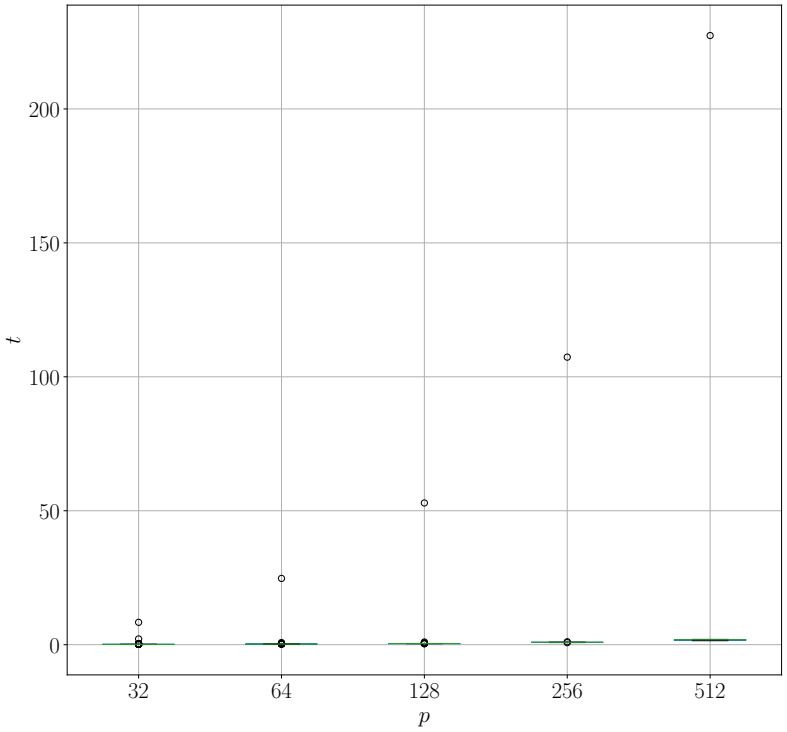
(e) $m = 512$



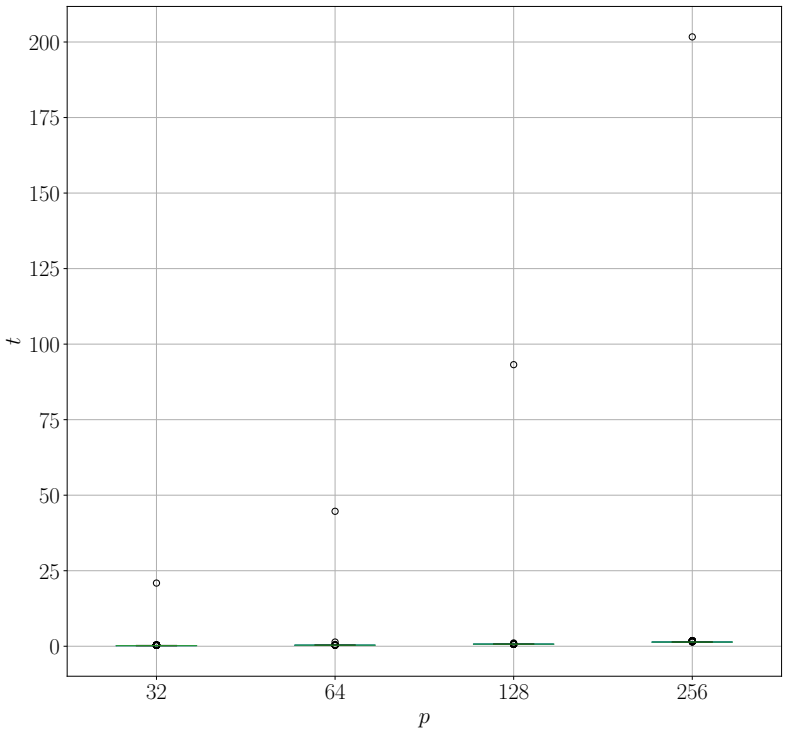
(f) $m = 1024$



(g) $m = 2048$

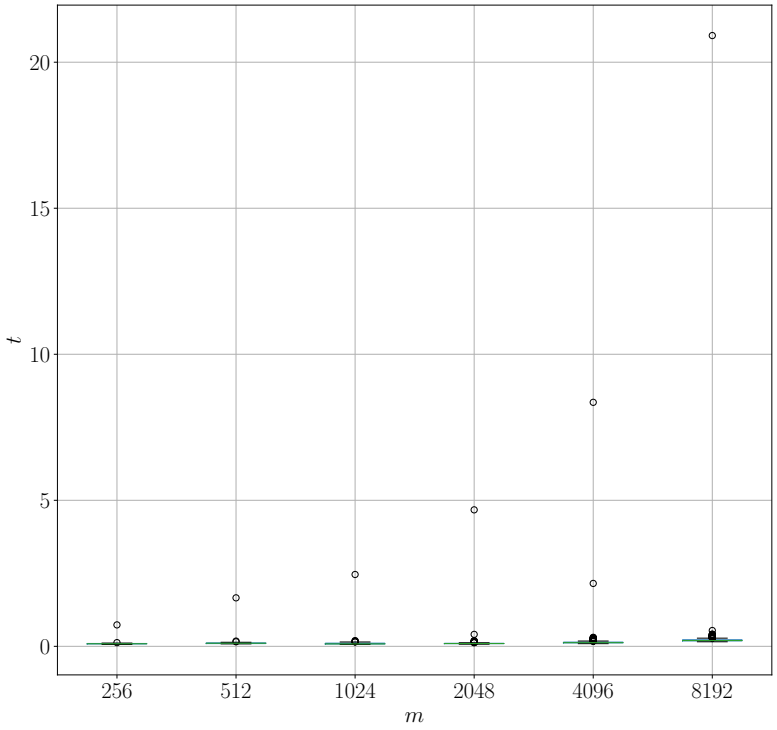


(h) $m = 4096$

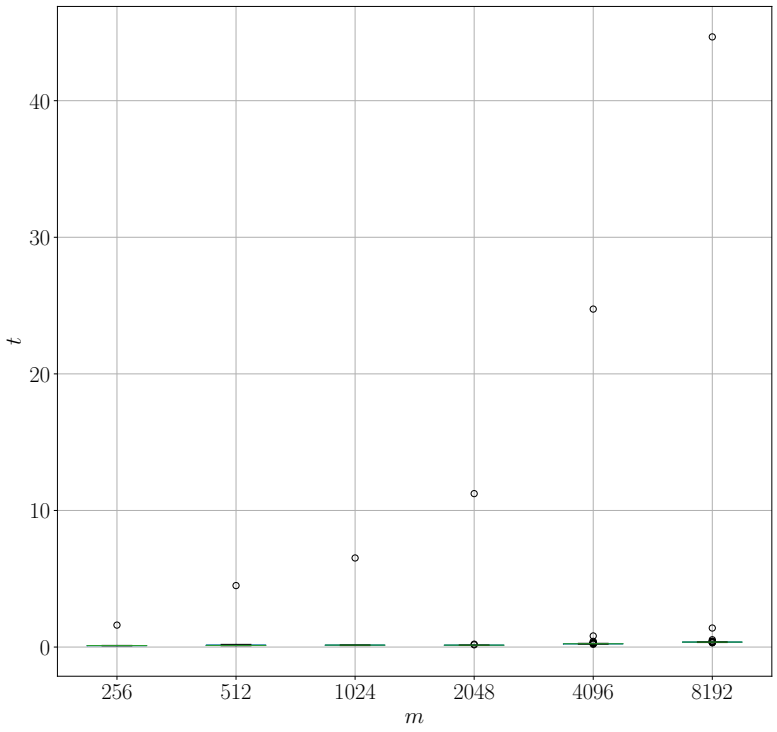


(i) $m = 8192$

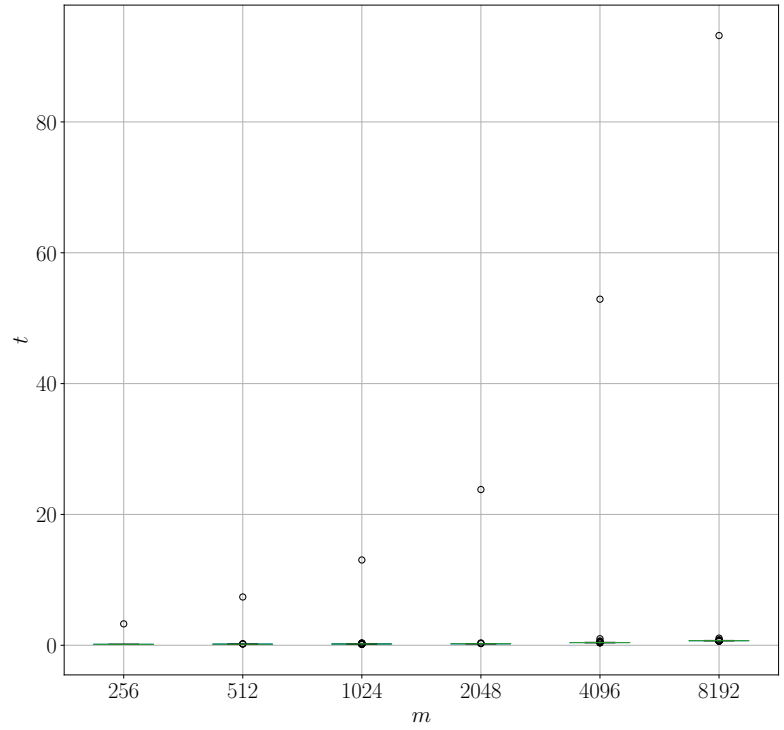
Figura 5.2: Tiempos de consulta variando p y dejando fija a m


20
15
10
5
0
t
256
512
1024
2048
4096
8192
m

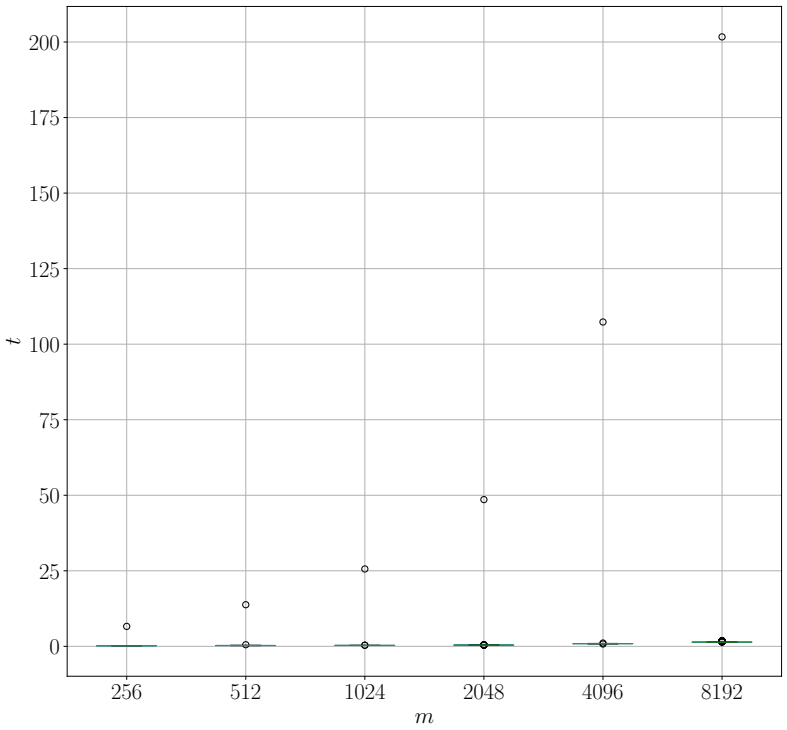
(a) $p = 32$


40
30
20
10
0
t
256
512
1024
2048
4096
8192
m

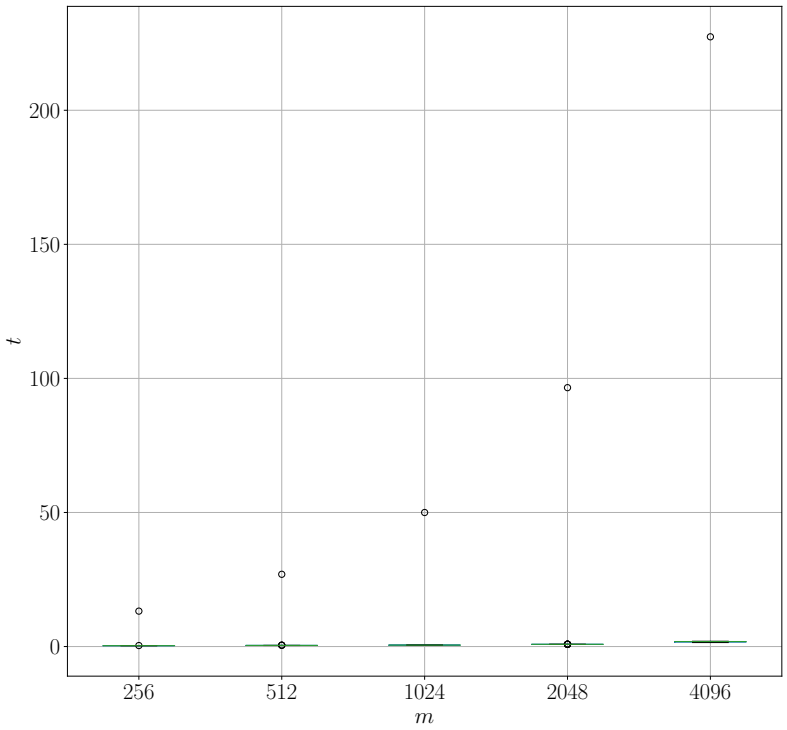
(b) $p = 64$


80
60
40
20
0
t
256
512
1024
2048
4096
8192
m

(c) $p = 128$


200
175
150
125
100
75
50
25
0
t
256
512
1024
2048
4096
8192
m

(d) $p = 256$



(e) $p = 512$

Figura 5.3: Tiempos de consulta variando m y dejando fija a p.

5.3.1.3 *Discusión*

El método de consulta en colecciones de nubes de puntos presentado mostró tener un alto desempeño en precisión, recuperación y tiempo de consulta una vez que el índice `R*-Tree` ha sido construido. La construcción del índice `R*-Tree` es el paso más oneroso del proceso, pero puede ser sustituido por un índice espacial más eficiente para el propósito.

6

CONCLUSIONES

Los métodos de apareamiento de patrones de puntos bajo transformaciones de semejanza basados en invariantes hiperbólicas presentados en esta tesis son eficaces mientras se cumpla que el número de puntos del patrón de consulta sea suficientemente grande como para que se recupere al menos una espiga del patrón a consultar.
Estos métodos se apoyan fuertemente en índices espaciales. De hecho, los pasos más costosos son los de creación y consulta de los índices espaciales. Esto permite aprovechar el amplio desarrollo existente en estos índices, así como cualquier progreso futuro en el área.
En comparación con los métodos del capítulo 2, el presentado aquí no requiere considerar a todos los posibles apareamientos. Por otro lado, el método del capítulo 2 sí es eficaz con patrones de consulta pequeños.
De la misma manera que en el capítulo 2 el método aquí presentado no tiene las limitaciones de métodos iterativos como RANSAC respecto a la relación entre el número de puntos de los patrones, ni a su cercanía.

[1] Peyman. Afshani y Timothy M. Chan. «Optimal Halfspace Range Reporting in Three Dimensions». En: *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*. 0 vols. Proceedings. Society for Industrial and Applied Mathematics, 4 de ene. de 2009, págs. 180-186. ISBN: 978-0-89871-680-1. DOI: 10.1137/1.9781611973068.21. URL: https://epubs.siam.org/doi/10.1137/1.9781611973068.21 (visitado 08-01-2020).

[2] Timothy M. Chan y Konstantinos Tsakalidis. «Optimal Deterministic Algorithms for 2-d and 3-d Shallow Cuttings». En: *31st International Symposium on Computational Geometry (SoCG 2015)*. Ed. por Lars Arge y János Pach. Vol. 34. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015, págs. 719-732. ISBN: 978-3-939897-83-5. DOI: 10.4230/LIPIcs.SOCG.2015.719. URL: http://drops.dagstuhl.de/opus/volltexte/2015/5135.

[3] Edgar Chávez, Ana C. Chávez Cáliz y Jorge L. López-López. «Affine Invariants of Generalized Polygons and Matching under Affine Transformations». En: *Computational Geometry* 58 (oct. de 2016), págs. 60-69. ISSN: 09257721. DOI: 10.1016/j.comgeo.2016.06.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0925772116300621 (visitado 29-06-2019).

[4] *Geopandas/Geopandas*. URL: https://github.com/geopandas/geopandas (visitado 21-11-2019).

[5] Christopher D. Manning, Prabhakar Raghavan e Hinrich Schütze. *Introduction to Information Retrieval*. USA: Cambridge University Press, 2008. ISBN: 0-521-86571-9.

[6] Apostolos N. Papadopoulos, Antonio Corral, Alexandros Nanopoulos y Yannis Theodoridis. «R-Tree (and Family)». En: *Encyclopedia of Database Systems*. Ed. por LING LIU y M. TAMER ÖZSU. Boston, MA: Springer US, 2009, págs. 2453-2459. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_300. URL: https://doi.org/10.1007/978-0-387-39940-9_300 (visitado 15-11-2019).

[7] Fabio Porto, João N. Rittmeyer, Eduardo Ogasawara, Alberto Krone-Martins, Patrick Valduriez y Dennis Shasha. «Point Pattern Search in Big Data». En: *Proceedings of the 30th International Conference on Scientific and Statistical Database Management* (Bozen-Bolzano, Italy). SSDBM '18. New York, NY, USA: ACM, 2018, 21:1-21:12. ISBN: 978-1-4503-6505-5. DOI: 10.1145/3221269.3221294. URL: http://doi.acm.org/10.1145/3221269.3221294 (visitado 07-05-2019).

[8] Kenneth H. Rosen, ed. *Handbook of Discrete and Combinatorial Mathematics*. 13. print. Boca Raton, Fla.: CRC Press, 2000. 1232 págs. ISBN: 978-0-8493-0149-0.

[9] Hanan Samet. *The Design and Analysis of Spatial Data Structures*. USA: Addison-Wesley Longman Publishing Co., Inc., 1990. ISBN: 0-201-50255-0.

[10] Masaharu Tanemura. «Statistical Distributions of Poisson Voronoi Cells in Two and Three Dimensions». En: *Forma* 18 (1 de ene. de 2003), págs. 221-247.