移动对象索引技术研究进展*)

廖 巍 熊 伟 景 宁 钟志农
(国防科技大学电子科学与工程学院 长沙 410073)

摘 要 在位置服务、交通控制等移动计算领域,移动对象索引技术广泛应用于对移动终端的空间位置进行存储和检索。本文深入分析了移动对象历史轨迹、当前位置和未来位置预测等各种索引技术,并根据索引空间及结构的不同对现有移动对象索引方法进行了详细的分类比较与讨论,对移动对象索引技术研究方向进行了展望。

关键词 移动对象索引,移动对象数据库,R 树,TPR 树

Research on Indexing Moving Objects Methods

LIAO Wei XIONG Wei JING Ning ZHONG Zhi-Nong
(College of Electronic Science and Engineering, National University of Deference Technology, Changsha 410073)

Abstract In the fields of mobile computing such as location-based services and traffic control, moving object indexing techniques are used to store and query the location of moving objects. This paper thoroughly analyzes indexing methods for historical trajectory, current position and future location of moving objects. And most existing moving objects indexing methods are classified and discussed thoroughly according to their indexing space and structures, and finally some open researching problems and interesting direction are presented.

Keywords Moving object indexing methods, MOD, R-tree, TPR-tree

在过去的二十年里,时空数据库研究领域得到了人们广泛的关注,在时空数据模型、时空数据索引与查询、时空推理及时空数据库体系结构设计等方面都出现了大量的研究成果。但总的来说,目前时空数据库在理论研究和应用上都处于不成熟的阶段,而且随着应用领域的不断扩展和新技术的出现,尤其是随着移动计算,无线通信及定位技术的发展,如何有效地对移动对象进行查询、管理以及提供准确的基于位置服务等应用需求使得时空数据库研究面临着新的挑战[1]。

移动对象的存储和检索技术根据查询时间窗口范围可以分为三类:检索移动对象历史轨迹信息、检索移动对象当前位置信息以及移动对象未来趋势预测。目前国内外许多学者针对不同的应用提出了各种移动对象索引技术,但总体来说移动对象索引技术方面的研究仍然处于初步阶段。其中移动对象未来趋势预测技术由于更适合在智能交通调度、基于位置服务等领域的应用而得到了广泛的关注,是一个充满挑战性的研究方向[3]。

本文第 1 节介绍移动对象历史轨迹的索引技术;第 2 节则描述了移动对象当前位置索引技术;第 3 节对当前移动对象索引研究热点未来轨迹预测技术进行了详细的介绍。最后对移动对象索引技术作了总结和展望。

1 移动对象历史轨迹索引

移动对象历史轨迹随着时间不断延伸,假若在数据库中存储移动对象的实际位置,那么当移动对象位置连续变化时,必须定时的向数据库发出更新请求,而数据库则记下每次更新时移动对象的空间位置。显然,利用这种简单的快照数据模型来描述移动对象历史轨迹,会造成数据库存储随着时间变化线性增长,在许多需要保存长时间历史轨迹的应用中是不可行的。为了减少数据库中移动对象历史轨迹的空间存储开销,人们提出了两种解决方法[2]:1)抽样方式,将移动对象历史轨迹按照某种采样时间间隔(或在特殊时间片上)进行抽样存储,然后对抽样点之间的轨迹则用线性插值进行拟合;2)存储移动对象的运动行为,即在数据库中存储描述移动对象运动的参量信息(如起始位置、速度矢量等),只有当移动对象的某个运动参量发生变化时才对数据库进行更新。根据上述思路研究者提出了许多索引方法,具体来说,移动对象历史轨迹检索方法可以分为以下三类:基于传统空间索引方式[4,5];基于重叠与多版本结构索引方式[6,7]及面向移动对象轨迹索引方式[8,9]。下面将分别予以详细介绍。

RT 树(RT-tree)[4]对标准 R 树进行扩充,利用时间间隔来标识移动对象历史轨迹有效时间范围,历史轨迹所覆盖的空间区域仍然使用 MBR 来近似。RT 树的节点记录形式为四元组模型(id, MBR, t_s, t_e)。其中 id 标识移动对象,MRB 是包含移动对象历史轨迹的最小包围框,t_s 和 t_e 表示移动对象历史轨迹 MRB 的有效时间间隔。RT 树的空间查询性能与标准 R-tree 相当,时态查询(包括时间片查询和时间窗口查询)则使用 TSB-tree 来进行访问,对于时间片查询具有较好的查询性能,但时间窗口查询在某些情况下则可能需要搜索整个 RT 树。

Theodoridis 等人则将时间维与空间维(二维空间)查询语义等同考虑,对 2D R 树在三维空间中进行扩展提出了 3D R 树(3D R-tree)[5]索引结构。3D R 树可以直接利用现有的 R 树算法来处理空间及时态查询而无需修改,其前提是必须知道移动对象历史轨迹存在的有效时间范围,如果移动对象

*)Supported by the High-Tech Research and Development Plan of China under Grant No. 2002AA104220; No. 2002AA134010; No. 2003AA5110 (国家高技术研究发展计划(863))。廖 巍 博士生,主要研究领域为空间数据库,时空数据库。

轨迹结束时刻不确定，那么 3D R 树将无法有效工作。比如若移动对象运动轨迹从某个历史时刻开始一直延伸到当前时刻(now)，显然包围移动对象轨迹的 3D MBR 将会非常巨大且造成空间区域的大量重叠，直接导致 3D R 树的查询搜索性能下降。因此，在应用中要求移动对象轨迹是封闭的(closed)，即移动对象历史轨迹必须在某个过去时刻终止而不能延伸到当前时刻。由于 3D R 树将时间维与空间维等同索引，时间片查询则必须扫描整个 3D R 树而非在此时间片下的有效子树，因此性能非常低下。

HR 树(HR-tree)[6]目的是避免在每个时间片上构建独立的 R 树而造成额外的空间存储开销。HR 树在每个时间片都有一个独立的 R 树根节点，包含当前的时间戳。相邻时间片的 R 树之间若存在公共对象(节点)时，则在新时间片 R 树节点中使用一个指针指向它们的公共节点而无需进行拷贝。由于相邻时间片 R 树公共节点只存储了一次，存储开销大大降低。显然，沿着不同的根节点可以访问各时间片下的移动对象。这种索引方式在回答时间片查询时非常有效，但是对时间窗口查询来说性能较差，而且可能会造成索引树中记录的大量复制。考虑这种情况，若相邻时间片的 R 树中只有一个移动对象位置发生变化，那么由于 MBR 变化所产生的传播操作将会导致包含此对象的叶节点(中间节点)中所有记录必须进行复制。

Tao 等人提出的 MV3R 树(MV3R-tree)[7]采用一种混合索引结构来处理移动对象历史轨迹查询。MV3R 树利用 MVR 树(MVR-tree)来处理时间片查询，对于时间窗口查询则使用一个建于叶节点之上的 3D R 树来处理。对于较短的时间窗口查询则根据优化准则从中选择合适的索引 MVR 树或 3D R 树来进行查询。MV3R 树以较高的空间存储代价换取了很好的时间片和时间窗口查询性能，是目前最高效实用的移动对象历史轨迹索引方法。

SETI[8](Scalable and Efficient Trajectory Index)将空间区域分割成静态且不重叠的分区，在每个分区下对移动对象的轨迹线段利用 R 树进行索引。其考虑是，相对于无限连续变化的时间维而言，移动对象轨迹受空间范围所限，那么可以把受限的空间区域利用某种规则静态划分以分而治之。良好的划分函数应尽量把同一移动对象不同轨迹线段聚类在同一个分区中。若一条轨迹线段穿过多个分区，那么必须将此线段裁剪并分别存储在不同分区 R 树中，这样会导致查询结果的重复，在查询处理后必须进行重复结果的剔出。相应地，时态查询则被转换为对折线段集合的空间窗口查询，利用几何计算方法可以得到空间窗口内的折线段集合，其对应的移动对象集合即是时态查询结果。

SEB 树[9]思想与 SETI 类似，但是在进行空间静态划分时允许不同分区之间区域重叠。对每个移动对象 SEB 树索引方法首先利用哈希变换将其映射到不同的分区中，然后在每个分区中利用 SEB 树对移动对象轨迹线段起始和结束位置进行索引。SEB 树与 SETI 索引最主要的区别在于 SEB 树存储移动对象轨迹线段的起始和终止位置(点)而非轨迹本身(线段)。SEB 树时态查询则被转换为对点集合的空间窗口查询，落在窗口内的移动对象轨迹线段起始和终止位置点所对应的移动对象即为满足时态查询的结果。

表 1 对目前移动对象历史轨迹索引技术进行了比较，从表中可以看出基于重叠和多版本的 MV3R-tree 和面向轨迹的 SEB-tree 查询性能较好，其中前者使用 R-tree 索引结构具有更好的通用性而得到了广泛的应用。

表 1　移动对象历史轨迹索引技术研究进展与比较

year	Technique	Time slice query	Time window query	Space Req.
1990	RT-tree	较好	较差	较少
1996	3D R-tree	差	较好	较少
1998	HR-tree	好	较好	多
2001	MV3R-tree	好	好	较多
2003	SETI	好	较好	较多
2003	SEB-tree	好	较好	较少

2　移动对象当前位置索引方法

在移动对象历史轨迹索引中，必须预先知道移动对象轨迹范围，即数据库存储移动对象的封闭轨迹，当前位置并不存储和检索，然而在现实许多应用中要求能够有效地对移动对象当前位置进行管理和查询。相对来说，检索移动对象当前位置更具有挑战性，其关键在于如何提高传统空间索引的频繁更新动态性能以满足动态环境下的应用要求。近年来针对移动对象当前位置查询所提出的索引的方法包括 2+3 R 树(2+3 R-tree)[10]、LUR 树(Lazy Update R-tree)[11]、自底向上更新策略(Bottom-up Update)[12]、哈希方法(Hash)[13]等。

2+3 R 树(2+3 R-tree)[10]通过维护两个 R 树索引来同时索引移动对象的当前位置和历史轨迹，可以回答移动对象历史轨迹查询和当前位置查询。其中，2D R 树索引用于检索移动对象当前位置，3D R 树索引用以存储移动对象三维历史轨迹(二维空间与时间维)。当移动对象位置发生更新时，算法则构建移动对象历史轨迹三维 MBR 并插入到 3D R 树中，然后在 2D R 树中更新移动对象当前位置。显然，2+3 R 树索引需要至少双倍的存储空间，其动态更新代价非常高昂，且维护两个 R 树之间的一致性也非常的困难，远远不能满足实际中应用。

使用 R 树来检索移动对象当前位置面临着索引树频繁更新的问题，标准 R 树更新算法采用删除-重插机制来实现移动对象位置更新，频繁更新会导致 R 树性能急遽下降。为了支持 R 树的频繁更新操作，文[11]提出了一种延迟更新(Lazy Update)策略，即若移动对象位置未超出当前节点 MBR，算法仅仅更新移动对象所在的数据页面并维持索引页面不变；对于移出 MBR 之外的移动对象，根据其超出 MBR 程度大小，判断是否将 MBR 进行有限扩展以包含移动对象新位置；或者利用标准 R 树更新算法进行更新以尽量减少位置更新所带来的索引树更新操作。自底向上更新(Bottom-up Update)[12]算法扩展了 LUR 树的思想，提出了新的自底向上更新策略。算法依据移动对象位置变化的程度判断是将其所在 MBR 进行有限扩展；或是将其移至某个兄弟节点中去；或采用标准 R 树删除-重插机制将移动对象插入到合适子树中。同时为了避免在进行自底向上更新时搜索父节点和兄弟节点所带来的额外磁盘 I/O，引入了一个紧凑的内存结构来直接访问 R 树的中间节点。自底向上更新算法是目前最有效的基于 R 树移动对象当前位置索引方法。

Song 等人[13]利用空间划分的思想来索引移动对象当前位置，首先静态的把空间划分为可以相互重叠的区域，然后根据当前位置将移动对象哈希映射到不同区域中去。只有当移动对象位置超出了当前区域才会更新此对象的索引记录，数据库中保存了移动对象位置近似视图。为了解决位置的不确

定性，在数据库和移动对象之间引入了一个过滤层用以保存移动对象的确切位置。相应地，范围查询被映射到不同区域中去进行，若一个区域完全被查询范围包含，那么返回区域中所有移动对象记录；若此区域与查询范围相交，那么区域中的所有记录必须返回到过滤层精确判断其是否满足查询谓词。

表2比较了移动对象当前位置的索引技术，其中支持Bottom-up Update算法的R-tree索引是目前具有最好查询和动态更新性能的索引方法。

表2 移动对象当前位置索引技术研究进展与比较

year	Technique	Query Performance.	Dynamic Performance.	Space Req.
1999	2+3 R-tree	好	较差	多
2002	LUR-tree	较好	好	较少
2003	Bottom-up Update	好	好	较少
2001	Hash	较好	较好	较多

3 移动对象当前和未来轨迹查询

在移动计算、位置服务等新兴应用中，需要对移动对象当前及未来位置预测以提供相关服务，针对移动对象当前和未来轨迹的索引方法是目前研究的热点。为了能够预测移动对象未来轨迹，需要存储一些额外的信息（如移动对象当前速度和目的地等）。然后根据移动对象当前的运动特性对未来轨迹进行预测，目前通常使用线性方程来描述移动对象的运动特性。在D维空间下，移动对象运动特性可以使用参考时间t_{ref}时位置$\overset{\rho}{x}_{ref}=(x_1,x_2,\cdots,x_d)$以及速度矢量$\overset{\rho}{v}=(v_1,\cdots v_2,\cdots,v_d)$来描述。移动对象在任何时间点$t(t\geqslant t_{ref})$的位置可利用公式$\overset{\rho}{x}_t=\overset{\rho}{x}_{ref}+\overset{\rho}{v}(t-t_{ref})$计算得出。在一维情况下移动对象轨迹表示为线性方程$x_t=at+b$，其中$a$和$b$都是常量。$a$表示移动对象运动速度，$b$表示移动对象参考位置，此时$t_{ref}=0$。移动对象当前和未来轨迹索引方法主要分为以下几类：在原始空间时态域中索引[14]、在变换域中索引[2,15,16]以及参数化的空间索引方式[17~20]。其中在变换域中索引和参数化的空间索引方式是目前研究的主要方向。

在二维空间中描绘运动方程$x_t=at+b$，令水平轴表示时间t，垂直轴表示移动对象位置x_t，我们可以得到二维空间中的线段集合。移动对象未来位置索引问题就转变为如何索引二维空间中的线段集合，从而可以利用现有的传统空间访问方法。目前公开研究成果中只有Tayeb等人提出的PMR四叉树（PMR-quadtree）[14]采用这种方式。其思想非常直观，当移动对象发出位置更新请求时，整个索引树被抛弃，然后使用新的位置信息重建索引树。为了避免由于移动对象位置频繁更新所带来的经常性索引树重构，PMR四叉树采用周期性重构策略，其时间间隔为ΔT。从抽象层次上将无限的时间范围分割为等间隔ΔT的时间片，在每个时间片下根据运动参量重构PMR四叉树，由于空间存储的限制，实际上只保存了当前时间片下的PMR四叉树。

总的来说，使用传统空间索引方式对移动对象未来轨迹进行检索存在两个主要缺点：1）使用最小包围框来拟合移动对象轨迹造成大量的死区存在；2）所有移动对象轨迹具有相同结束时间戳造成MBR分布的极不均匀。由于在原始空间时态域中对移动对象未来位置进行索引的困难性，人们考虑利用变换方式在二元空间中进行索引和管理。

Kollios等人采用二元变换（Duality transformation）[15]将时空域中的线段（移动对象轨迹）变换为二维空间中的一个点来进行索引，即把原始空间时态域中的线性运动方程$x_t=at+b$表示为二维变换空间中的一个点(a,b)，其中速度大小a表示水平轴方向，参考位置b表示垂直轴方向。由于二元变换空间中数据分布的不均匀性，二元变换空间点使用k-d树来进行索引。相应地，原始空间时态域中的窗口查询则被映射为二元空间的多边形区域查询。Chon等人[2]使用一个四元组(s,e,t,v)来描述移动对象运动，其中s表示移动对象起始位置，e表示移动对象目的位置，t表示移动对象运动起始时刻，v表示移动对象初始速度。假定移动对象运动模型四个参量中只有两个参量可以自由变化，那么就存在六种模型可以描述移动对象运动。在这六种组合中，作者认为起始位置s和初始速度v固定的运动模型即S-V模型（S-V model）符合现实应用且最为有效。S-V模型含义即是指固定参量起始位置s和初始速度v，在二元空间中使用起始时间t和移动对象目的位置e分别作为水平轴和垂直轴。同样原始空间时态域中窗口查询则被转换为二元空间中多边形查询，并使用SS-tree来进行索引。由于S-V模型限定速度为常量，在公路交通控制系统、飞机调度中有较好的应用。

STRIPE索引[16]采用类似的变换方式将$d+1$维的原始时空域（d维空间域和一维时间域）转换到$2d$维的变换空间中去。对一维原始空间来讲，其二元变换空间以速度为垂直轴，移动对象位置为水平轴。通过引入最大速度矢量$\overset{\rho}{v}_{max}$和最大位置矢量$\overset{\rho}{p}_{max}$将二元空间索引范围局限于$[0,2\overset{\rho}{v}_{max}]$与$[0,2\overset{\rho}{p}_{max}]$之间。二元变换空间中使用PR四叉树对变换点进行索引，即把d维的$\{(v_1,p_1),(v_2,p_2),\cdots,(v_d,p_d)\}$每个二元平面等分成四份。每次划分将变换空间分割为$2^{2d}=4^d$个矩形区域，即PR四叉树的每个非叶节点扇出为$2^{2d}$。STRIPE使用两个PR四叉树索引交替更新回答查询，每个索引树有效时间为其生命期L。每隔L时间，其中一个索引被删除重建，而查询则交替到另外一个索引树上进行。如此循环以保证查询和更新效率，其代价是占用双倍的磁盘存储空间。

目前移动对象未来轨迹索引研究趋势是利用参数化包围框在原始时空域中对移动对象位置进行索引，即利用具有时间函数的包围框将移动对象始终包含在同一个时间参数化矩形框内，那么通过时间函数，在任何时间片下的R树索引结构都可以计算得出，空间查询则基于此时的索引进行处理。现有的参数化空间访问方法大都基于R树结构，如TPR树（time-parametered R-tree）及其变种TPR*树、VCI R树（VCI R-tree）、STP树（STP-tree）等。

TPR树（TPR-tree）[17]使用时间参数化包围框来包含移动对象，其索引结构与R树类似。但TPR树采用谨慎的BR策略，在某些时刻（如构建时刻）包围框是最小的，但在其后的时间内往往不是。从一维角度考虑，MBR最（大）小边界速度为其范围内所有点的最（大）小速度，从而保证所有点始终包含在同一个MBR中。TPR树索引节点记录不仅存储了移动对象（子树）MBR，而且包含了移动对象（子树）MBR速度矢量。由于节点MBR呈现出一种随时间变化的动态性，随着时间延伸包围框会越来越大，导致空间区域的大量重叠从而使得性能下降。因此，必须在适当时候对TPR树进行重构以提高查询性能。TPR树删除、插入及更新等动态操作采用

R^* 树标准算法，但使用时间参数化的度量准则，其表达形式如公式 $\int_{T0}^{T0+L} M(t)\,dt$，其中 $M(t)$ 是 R^* 树中对应的度量（如面积、周长等），L 是索引的生命期。Tao 等人提出的 TPR* 树（TPR*-tree）[18]修改了 TPR 树的动态操作算法。通过维护一个代价下降优先队列保证插入路径选择的全局最优，保持 TPR 树 MBR 的紧致性以提高查询性能。对于上溢节点的分裂算法，TPR* 树只是在必要的时候才对节点分裂并重插。以二维空间移动对象为例，节点首先在所有可能的 8 个方向（4×d）上进行排序，并从中选择一种最好的分裂方式，实际上此时节点并不分裂，而是把节点中 30% 的记录删除后重插。如果在重插过程中发生上溢，那么就直接进行节点分裂以避免传递操作。TPR* 树是目前最好的参数化空间访问方法。

与 TPR 树参数化描述方式不同，VCI R 树（VCI R-tree）[19]通过最大速度来扩展 MBR 使其始终包含所有子节点。VCI R 树节点记录中加入了最大速度域 v_{max}，v_{max} 是索引构建时节点 MBR 内所有子节点 MBR（移动对象 MBR）最大速度。VCI R 树并不需要知道每个移动对象的速度（只要其速度不超过最大速度），从而可以避免由于速度变化造成的索引频繁更新。VCI R 树将查询看成“数据”，数据看成“查询”，并在“数据”上建立索引称之为查询索引（query index）。移动对象必须周期性或不定时（位置变化超出阈值等）报告运动参量以更新“查询”。相应地，连续查询则被这种周期性或不定时的查询更新所代替，并基于查询索引增量计算查询结果。这种方式在处理多重连续查询时非常有效。

表 3　移动对象未来轨迹索引技术研究进展和比较

year	Technique	Time slice query	Time window query	Space Req.
1998	PMR quadtree	较差	差	较多
1999	Duality transform	较好	较好	一般
2001	S-V model	较好	较好	一般
2004	STRIPE	好	较好	较多
2000	TPR-tree	较好	较好	一般
2003	TPR*-tree	好	好	一般
2002	VCI R-tree	较好	较好	一般
2004	STP-tree	好	较好	较多

Tao 等人在文[20]中提出了一种能够处理具有未知运动模式移动对象轨迹查询的时空查询模型及 STP 树索引结构。其思想是在服务器端和客户端使用不同的轨迹函数来描述移动对象的运动。服务器端对所有移动对象采用同样的轨迹函数进行拟合，而在客户端上维持一个精确轨迹模型。当移动对象在服务器上的拟合轨迹误差超出阈值时，才会对索引中轨迹函数参量进行更新。查询首先在服务器上作粗略估算，然后在客户移动对象上作精确查询以得到准确结果。STP 树索引结构类似于 TPR 树，在节点记录中增加了轨迹参数矩阵、更新时刻和误差阈值（horizon bounds）。由于节点记录结构增大，导致 STP 树节点扇出大大降低，其实际查询性能只是稍好于 TPR 树而逊于 TPR* 树。

表 3 对目前移动对象未来轨迹预测技术中具有代表性的算法进行了比较，可以看出未来研究的趋势是变换方式和参数化方法，其中 TPR 树及变种 TPR* 树是目前最重要的移动对象未来位置参数化索引方法。

总结与展望　本文对面向位置服务等移动计算领域的移动对象索引技术进行了详细的分类与介绍，移动对象索引技术根据查询的时间范围（历史轨迹、当前位置、未来轨迹）分成三类。其中移动对象历史轨迹与当前位置索引技术已比较成熟，目前大部分研究工作已集中于移动对象未来轨迹索引技术方面。虽然有不少成果见于国内外文献，但现有的方法在实际应用中还存在着许多问题。

近年来的工作大都独立研究移动对象历史轨迹、当前位置与未来趋势预测的索引方法，在如何同时解决上述三种查询的索引方法上研究仍然欠缺，目前只有 Jimeng Sun 等人[21]在这方面进行了尝试性的工作。另外，现有的移动对象索引技术研究侧重于解决窗口查询、邻近查询，但是针对连接查询等方面的索引技术却研究甚少。值得注意的是，针对大量并发查询处理的移动对象索引方法研究比较少，但在实际应用中往往会面对这些问题。随着移动计算、位置服务技术等应用领域的发展，新兴应用的出现将会不断推动移动对象索引技术新问题和新方向的出现。

参 考 文 献

1 König-Ries B, Makki K, et al. Research Direction for Developing an Infrastructure for Mobile & Wireless Systems. In: Consensus Report of the NSF Workshop, 2001
2 Mokbel M F, Ghanem T M, Aref W G. Spatio-temporal Access Methods. IEEE Data Engineering Bulletin, 2003
3 Chon H D, Agrawal D, Abbadi A E. Data Management for Moving Objects. In: Proceeding of International Telematics and LBS Workshop. 2003
4 Xu X, Han J, Lu W. RT-Tree: An Improved R-Tree Indexing Structure for Temporal Spatial Databases. In: Proc. of the Intl. Symp. on Spatial Data Handling, SDH, 1990
5 Theodoridis Y, Vazirgiannis M, Sellis T. Spatio-Temporal Indexing for Large Multimedia Applications. In: Proc. of the IEEE Conference on Multimedia Computing and Systems, ICMCS, 1996
6 Nascimento M A, Silva J R O. Towards historical R-trees. In: Proc. of the ACM Symp. on Applied Computing, SAC. 1998
7 Tao Y, Papadias D. MV3R-Tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries. In: Proc. of VLDB 2001
8 Chakka V P, Everspaugh A, Patel J M. Indexing Large Trajectory Data Sets with SETI. In: Proc. of CIDR, 2003
9 Song Z, Roussopoulos N. SEB-tree: An Approach to Index Continuously Moving Objects. In: Proc. of MDM, 2003
10 Nascimento M A, Silva J R O, Theodoridis Y. Evaluation of Access Structures for Discretely Moving Points. In: Proc. of STDBM, 1999
11 Kwon D, Lee S, Lee S. Indexing the Current Positions of Moving Objects Using the Lazy Update R-tree. In: Proc. of MDM, 2002
12 Lee M, Hsu W, Jensen C, Cui B, Teo K. Supporting Frequent Updates in R-Trees: A Bottom-Up Approach. In: Proc. of VLDB, 2003
13 Song Z, Roussopoulos N. Hashing Moving Objects. In: Proc. of MDM, 2001
14 Tayeb J, Ulusoy O, Wolfson O. A Quadtree-Based Dynamic Attribute Indexing Method. The Computer Journal, 1998
15 Kollios G, Gunopulos D, Tsotras V J. On Indexing Mobile Objects. In: Proc. of ACM PODS, 1999
16 Patel J M, Chen Y, Chakka V Y. STRIPES: An Efficient Index for Predicted Trajectories. ACM SIGMOD, 2004
17 Saltenis S, Jensen C S, Leutenegger S T, Lopez M A. Indexing the Positions of Continuously Moving Objects. In: Proc. of ACM SIGMOD, 2000
18 Tao Y, Papadias D, Sun J. The TPR*-Tree: An Optimized Spatio-Temporal Access Method for Predictive Queries. In: Proc. of VLDB, 2003
19 Prabhakar S, Xia Y, Kalashnikov D V, Aref W G, Hambrusch S E. Query Indexing and Velocity Constrained Indexing: Scalable Techniques for Continuous Queries on Moving Objects. IEEE Transactions on Computers, 2002
20 Tao Y, Faloutsos C, Papadias D, Liu B. Prediction and Indexing of Moving Objects with Unknown Motion Patterns. In: Proc. of ACM SIGMOD, 2004
21 Sun Jimeng, Papadias D, Tao Yufei, Liu Bin. Querying about the Past, the Present, and the Future in Spatio-Temporal Databases. In: Proc. of ICDE, 2004