Spatiotemporal data model for network time geographic analysis in the era of big data

Bi Yu Chen[a*], Hui Yuan[a,b], Qingquan Li[a,c], Shih-Lung Shaw[a,d], William H.K. Lam[e] and Xiaoling Chen[a]

[a]State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan; University, Wuhan, China; [b]School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China; [c]Shenzhen Key Laboratory of Spatial Smart Sensing and Services, Shenzhen University, Shenzhen, China; [d]Department of Geography, University of Tennessee, Knoxville, TN, USA; [e]Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract

There has been a resurgence of interest in time geography studies due to emerging spatiotemporal big data in urban environments. However, the rapid increase in the volume, diversity, and intensity of spatiotemporal data poses a significant challenge with respect to the representation and computation of time geographic entities and relations in road networks. To address this challenge, a spatiotemporal data model is proposed in this article. The proposed spatiotemporal data model is based on a compressed linear reference (CLR) technique to transform network time geographic entities in 3-dimensional (x, y, t) space to 2-dimensional CLR space. Using the proposed spatiotemporal data model, network time geographic entities can be stored and managed in classical spatial databases. Efficient spatial operations and index structures can be directly utilized to implement spatiotemporal operations and queries for network time geographic entities in CLR space. To validate the proposed spatiotemporal data model, a prototype system is developed using existing 2-dimensional GIS techniques. A case study is performed using large-scale datasets of space-time paths and prisms. The case study indicates that the proposed spatiotemporal data model is effective and efficient for storing, managing and querying large-scale datasets of network time geographic entities.

Keywords: Time geography; spatiotemporal data model; spatiotemporal query; spatiotemporal big data; compressed linear reference

1. Introduction

Time geography is a powerful framework for studying human activities under various space-time constraints. The framework provides a set of well-defined space-time entities (e.g. space-time path and prism) and relations (e.g. bundling and intersections) to analyze human activities in space and time. It has served as a fundamental analytical framework for many

[*] To cite this paper: Chen, B.Y., Yuan, H., Li, Q.Q., Shaw, S.-L., Lam, W.H.K. and Chen, X., 2016, Spatiotemporal data model for network time geographic analysis in the era of big data. International Journal of Geographical Information Science, 30, pp. 1041-1071.

scientific fields, such as transport geography and urban planning (Huang and Wu, 2008; Fang et al., 2011; Fang et al., 2012; Kwan, 2013; Versichele et al., 2014; Charleux, 2015).

The concept of time geography was first introduced by Torsten Hägerstrand in the 1960s (Hägerstrand, 1970). In early studies, time geography was qualitative and lacked formal and analytical statements of basic entities and relations (Long and Nelson, 2013). With the availability of geographical information science (GIS) tools, the last two decades have witnessed a resurgence of time geography. Research efforts have focused on establishing measurement theory and developing computational tools for quantitative time geographic analysis (Miller, 2005; Kwan and Weber, 2003). Recognizing that people in urban area are generally traveling within networks, researchers have extended the time geographic framework from planar space to constrained network space and considered the complexities of traffic conditions in networks (Miller, 1991; Neutens et al., 2008; Miller and Bridwell, 2009; Chen et al., 2013). Geo-visualization tools integrated with GIS platforms also have been developed to explore human activity and travel patterns (Kwan, 2000; Yu and Shaw, 2008; Shaw and Yu, 2009; Chen et al., 2011).

In recent years, there has been a wider resurgence of interest in time geography due to the increasing availability of spatiotemporal Big Data in urban environments. The recent development of information and communication technologies (ICTs) have made it technically and economically feasible to collect large-scale individual movement data. Fine gained movement data include taxi trajectories, mobile phone records, smart card data, social media check-ins, and various user generated geographic information (Li et al., 2013; Yue et al., 2014). Spatiotemporal Big Data offers an unprecedented opportunity for time geographic studies to uncover people's mobility patterns and their interactions with the urban environment (Richardson et al., 2013; Goodchild, 2013; Kwan and Neutens, 2014; Song et al., 2010; Pei et al., 2014). However, the rapid increase in the volume, diversity, and intensity of movement data poses a significant challenge to contemporary GIS platforms with respect to the representation and computation of time geographic entities and relations. To address this challenge, the present study proposes an effective spatiotemporal data model with capability to store, manage, and analyze large movement datasets and thereby support large-scale time geographic analysis.

The integrated representation of spatial and time is an active frontier in GIScience research. Many spatiotemporal data models have been proposed to represent discrete changes of geographical phenomena (e.g. landuse changes), including snapshot model (Armstrong, 1988), spatiotemporal composite model (Langran and Chrisman, 1988), event-based spatio-temporal data model (Peuquet and Duan, 1995), three-domain data model (Yuan, 1999), geo-atom model (Goodchild et al., 2007; Chen et al., 2012), object-oriental models (Wang and Cheng, 2001), amongst others. However, these spatiotemporal data models are not tailored to the representation of time geographic entities and relations.

Effective storing and management of movement data has also inspired a large amount of literature on moving objects databases. Research regarding moving objects databases started from the work of Güting (Erwig et al., 1999) and Wolfson (Wolfson et al., 1998). Effective designs of moving objects databases have been proposed for storing, indexing, and querying movement data in unconstrained planar space (Güting and Schneider 2005). Research efforts have recently focused on extending moving objects databases to constrained network space (Güting et al., 2006; de Almeida and Güting, 2005; Li and Lin, 2006; Xu and Güting, 2013; Popa et al., 2015). While current moving objects databases can offer efficient methods for storing and retrieving movement data, they seldom integrate with the GIS platforms. Moreover, current moving objects databases only model the space-time path entity, ignoring other time geographic entities (e.g. space-time prism) and relations (e.g. path-prism and prism-prism intersections). Therefore, current moving objects databases are not suitable for large-scale time geographic analysis.

To fill the gap, the present study proposes a spatiotemporal data model for large-scale time geographic analysis. The proposed spatiotemporal data model is based on a compressed linear reference (CLR) technique to transform network time geographic entities in (x, y, t) space to 2-dimensional (2D) CLR space. Using the proposed model, network time geographic entities (i.e. space-time paths, stations, prisms and lifelines) can be stored and managed in classical spatial databases. Efficient spatial operations and index structures can be directly utilized to implement spatiotemporal operations and queries for network time geographic entities in CLR space. To validate the proposed model, a prototype system is developed using existing 2D GIS techniques. A real-world case study is also performed using large-scale datasets of space-time paths and prisms. The case study indicates that the proposed spatiotemporal data model is effective and efficient for storing, managing, and querying large-scale datasets of network time geographic entities.

Section 2 briefly introduces time geographic entities and relations in the planar space (Miller, 2005) to provide the necessary background. Section 3 introduces the representation of network time geographic entities and relations in CLR space. Section 4 presents the proposed spatiotemporal data model, associated spatiotemporal database design, and spatiotemporal queries. Section 5 details a case study to illustrate the applicability of the proposed model. Section 6 concludes the article with possible directions for future research.

2. Time geographic entities in planar space

The space-time path and prism are two entities central to time geography. The *space-time path* (also called trajectory in the literature) traces the movement of an individual in space and time. As illustrated in Figure 1, it is represented by a space-time polyline in (x, y, t) space with two spatial axes for geographic space and a third orthogonal axis for time. The (x, y, t) space is also known as the space-time cube (Kwan, 2000; Nakaya, 2013). The space-time path P^q of the individual q comprises a set of control points and a corresponding sequence of path segments connecting these points (Miller, 2005). Typically, the control points are observation records collected by location aware technologies (LAT) such as GPS devices. Each control point, c_i, consists of the triple

$$c_i = < x_i, y_i, t_i >, \tag{1}$$

where x_i and y_i are spatial coordinates, and t_i is a time stamp. It is commonly assumed in the literature that the individual is moving at a constant speed between each two adjacent control points, c_i and c_j (Wolfson et al., 1998; Miller, 2005). Using this assumption, the path segment s_{ij} between c_i and c_j is a straight line

$$s_{ij} = < c_i, c_j > \tag{2}$$

The speed of path segment is

$$v_{ij} = D(x_i, y_i, x_j, y_j) / (t_j - t_i) \tag{3}$$

where $D(x_i, y_i, x_j, y_j)$ is the Euclidian distance between two locations. A vertical path segment indicates that the individual is stationary in space, perhaps participating in activities. The space-time path therefore can be expressed as a 3D *LineString* comprising a set of straight line segments

$$P^q = < s_{o1}, ..., s_{ij}, ..., s_{kd} > \tag{4}$$

The individual's location on the space-time path at any time stamp $t_k \in [t_i, t_j]$ can be determined from

$$P^q(t_k) = \{(x_k, y_k) \mid x_k = (1-\lambda)x_i + \lambda x_j; y_k = (1-\lambda)y_i + \lambda y_j; \lambda = (t_k - t_i)/(t_j - t_i)\} \tag{5}$$

In principle, representing the space-time path as straight line segments (i.e. the assumption of constant speed between two adjacent control points) can capture complete information of a trajectory by collecting every location with direction and speed changes. However, in practice, movement data are usually collected at regular or irregular time intervals, and the resolution of a space-time path depends on its sampling frequency.

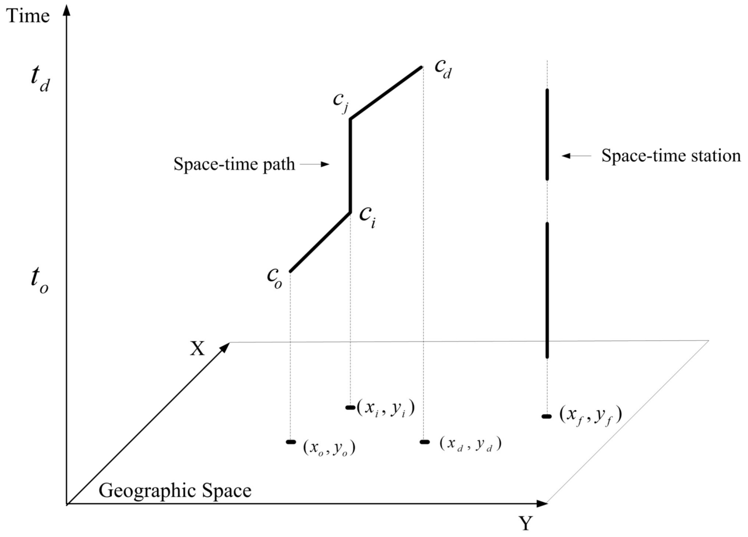


Figure 1. Space-time paths in planar space

The *space-time station*, Q^f, also illustrated in Figure 1, is a vertical space-time path

$$Q^f(t_k) = \{(x_k, y_k) \mid x_k = x_f; y_k = y_f; \forall t_k \in ([t_s^f, t_e^f], [t_{s'}^f, t_{e'}^f], \ldots)\} \tag{6}$$

It generally refers to an activity location, f (e.g. store, cinema, restaurant, etc.) where space-time paths can bundle or cluster in space and time. The finite durations of the station $([t_s^f, t_e^f], [t_{s'}^f, t_{e'}^f], \ldots)$ refer to time periods when the activity location is available (e.g. store operating hours).

Given an origin, $c_o = (x_o, y_o, t_o)$ and a destination $c_d = (x_d, y_d, t_d)$, *the space-time prism* (STP_{od}), delimits the space-time extent for an individual to conduct discretionary travel from c_o and reach c_d. Figure 2 illustrates this space-time prism in planar space. The space-time prism is commonly used for activity scheduling and accessibility analysis. The two control points (also called anchor points) generally obtained from activity diary data, represent an individual's fixed activities, such as home, work, or meetings. Between two fixed activities, a flexible activity may be scheduled by the individual at location (x, y) and time t with minimum duration $c_{\min}$. Let $T(x_o, y_o, x, y)$ be the minimum travel time from the origin (x_o, y_o) to location (x, y). The earliest arrival time at the flexible activity location can be simply calculated by

$$t^- = t_o + T(x_o, y_o, x, y) = t_o + D(x_o, y_o, x, y) / v_{\max} \tag{7}$$

where $v_{\max}$ is the maximum travel speed. To arrive at destination (x_d, y_d) on time, the latest departure time from location (x, y) can be calculated by

$$t^+ = t_d - T(x, y, x_d, y_d) = t_d - D(x, y, x_d, y_d) / v_{\max} \tag{8}$$

where $T(x, y, x_d, y_d)$ is the minimum travel time from (x, y) to the destination. This location is feasible for activity participation if the following constraint is satisfied:

$$t^{+}-t^{-}-c_{\min}=t_d-t_o-T(x_o,y_o,x,y)-T(x,y,x_d,y_d)-c_{\min}\geq 0 \tag{9}$$

Therefore, the potential path area (PPA) delimiting all potential locations for activity participations is defined as

$$\mathrm{PPA_{od}}=\{(x,y)\mid t^{+}-t^{-}\geq c_{\min}\} \tag{10}$$

The space-time prism is a 3D prism shaped structure capturing all potential space-time locations for an individual to conduct the flexible activity, and can be expressed as the intersection of a forward cone, a backward cone, and a cylinder:

$$\mathrm{STP_{od}}=<\mathrm{FC_o}\cap\mathrm{BC_d}\cap\mathrm{C_{od}}> \tag{11}$$

$$\mathrm{FC_o}=\{(x,y,t)\mid t^{-}\leq t,t\leq t_d\} \tag{12}$$

$$\mathrm{BC_d}=\{(x,y,t)\mid t^{+}\geq t,t\geq t_o\} \tag{13}$$

$$\mathrm{C_{od}}=\{(x,y,t)\mid t^{+}-t^{-}\geq c_{\min},t_o\leq t<t_d\} \tag{14}$$

where $\mathrm{FC_o}$ is the forward cone comprising all space-time locations that can be reached from origin c_o by elapsed time $t-t_o$; $\mathrm{BC_d}$ is the backward cone encompassing all space-time points where an individual can return to destination c_d in the remaining time budget t_d-t; and $\mathrm{C_{od}}$ is the cylinder delimiting all reachable geographic locations. The footprint of the 3D prism in geographical space is the PPA. The potential locations of the individual at a given time instance $t_k\in[t_o,t_d]$ are a subset of PPA,

$$STP_{od}(t_k)=\left\{(x_k,y_k)\mid\left(t_k^{-}\leq t_k\right)\wedge\left(t_k^{+}\geq t_k\right)\wedge\left(t_k^{+}-t_k^{-}\geq c_{\min}\right)\right\} \tag{15}$$

where $\wedge$ indicates the logical predicate AND. The height of the space-time prism at any location (x_k,y_k) is the time resource available for activity participations determined by the earliest arrival and latest departure times,

$$STP_{od}(x_k,y_k)=\{t_k\mid t_k^{-}\leq t_k\leq t_k^{+}\} \tag{16}$$

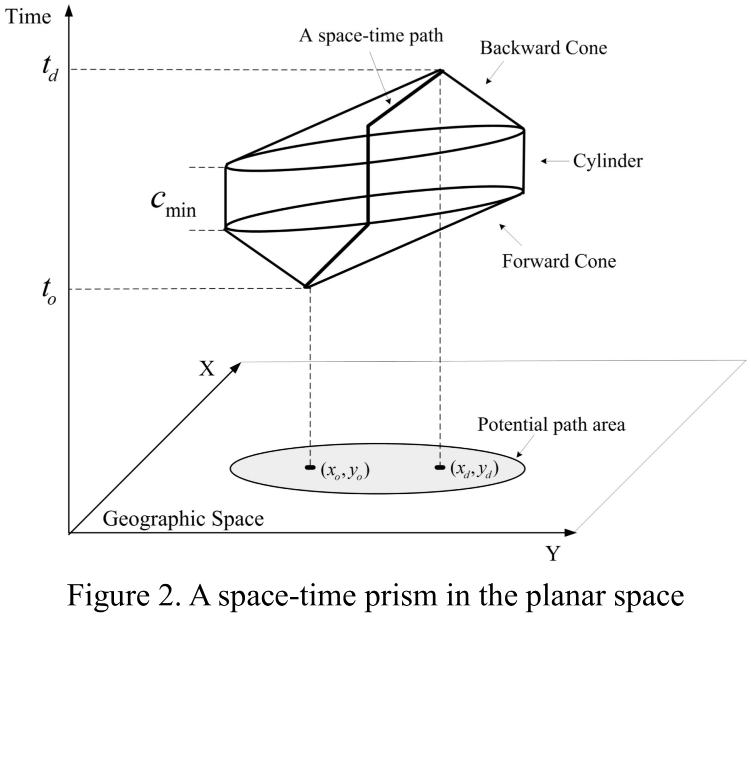


Figure 2. A space-time prism in the planar space

Figure 2 also shows the connection between the space-time path and prism. The space-time path represents a revealed movement of activity participation, while the space-time prism delimits all possible movements between two control points. Analyzing an individual's activities may require a combination of several space–time paths and prisms into a composite object, the *space-time lifeline* (Miller, 2005). Figure 3 illustrates a simple space-time lifeline with one space-time path and two space-time prisms. The space-time lifeline L^w is represented by the combination of space–time paths and prisms,

$$L^w = \{P^q, ...; STR_{od}, ...\} \tag{17}$$

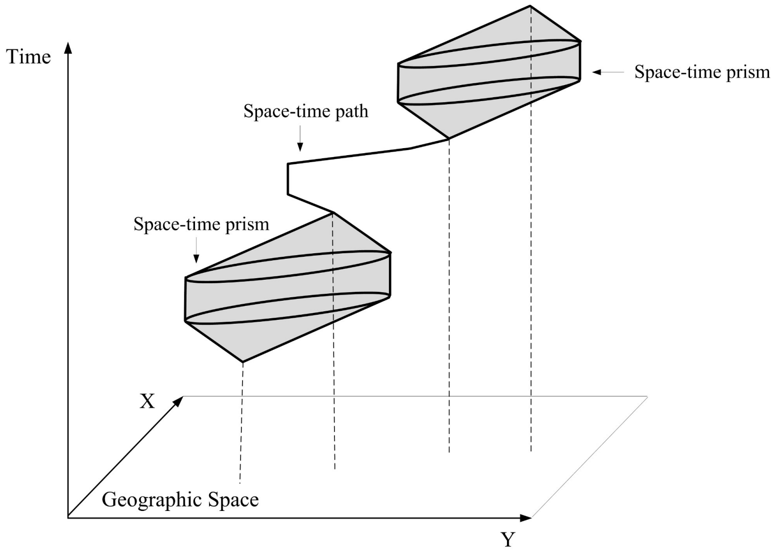


Figure 3. A space-time lifeline in the planar space

In addition to space-time entities, time geography defines two basic space-time relations, namely *intersections* and *bundles*. Intersections are where two or more time geographic entities co-exist in space and time. As shown in Figure 4, they can be classified into path-path, path-prism, and prism-prism intersection cases. Space-time intersections are important to time geographic analysis. For example, a joint activity cannot be scheduled for two entities unless their space-time prisms intersect.

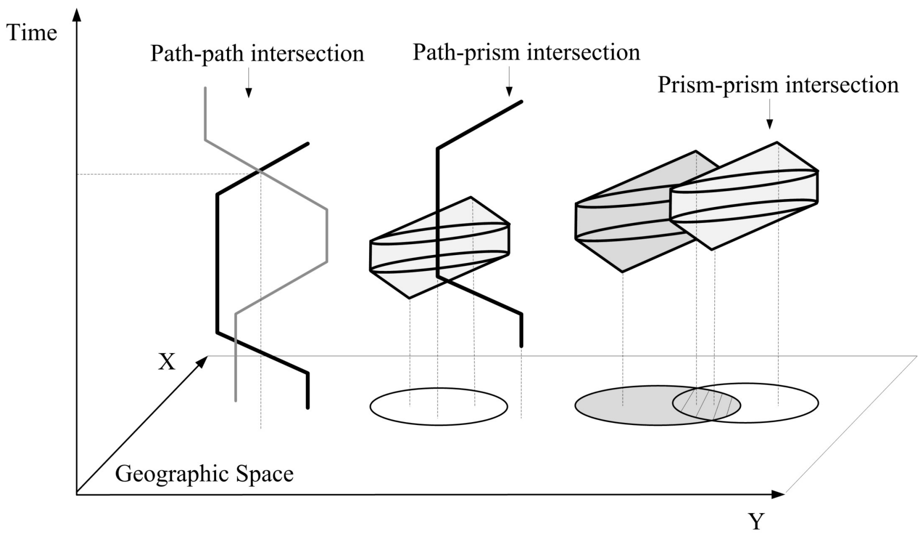


Figure 4. Space-time intersections

Bundles are path-path relations, with respect to the convergence of space-time paths for some shared activities. As illustrated in Figure 5, bundling of space-time paths does not require co-existence of paths in space and time but allows a spatial tolerance ω at any time instance. Path bundling is useful for monitoring crowd dynamics at large events or identifying an individual's activities from his multi-day paths.

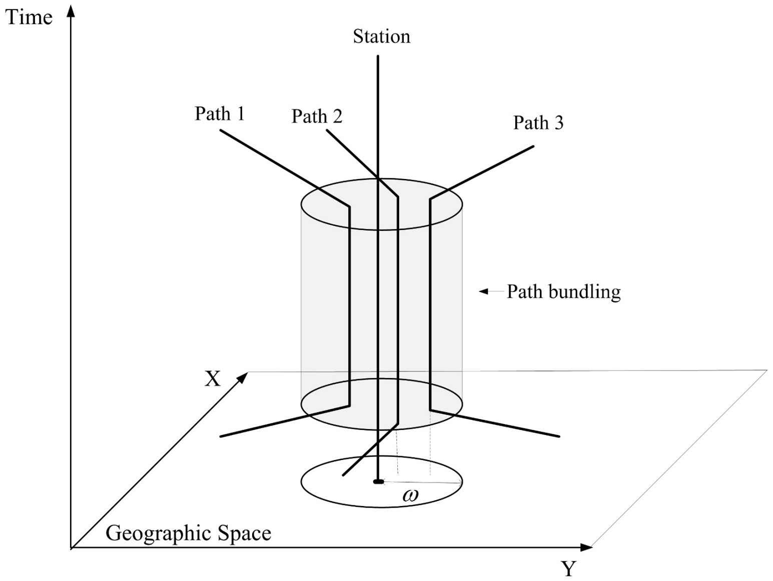


Figure 5. Space-time path bundling

The representation of time geographic entities by 3D lines and/or volumes in (x, y, t) space is rigorous and analytical (Miller, 2005). However, this representation technique poses significant challenges to contemporary 2D GIS and database platforms for efficiently storing, querying, and analyzing large-scale time geographic entities (Kwan and Neutens, 2014; Nakaya, 2013).

3. Representation of network time geographic entities and relations

People in urban areas do not move freely in (x, y, t) space, but rather within spatially embedded road networks. In this section we present a spatiotemporal data model for network time geographic entities and relations, which enables more efficient time geographic analysis. A road network can be represented as a directed graph, $\mathrm{G}=(\mathrm{N},\ \mathrm{A})$, comprising a set of nodes N, and a set of links A. Each link, $a_u \in A$, has a set of attributes including identity, l_u; length, d_u; and travel time, τ_u. To guarantee link uniqueness, link ID l_u is set as a unique positive integer number. A link, a_u, has a starting node, n_s^u; an ending node, n_e^u; and a set of intermediate vertices, n_i^u. The link geometry is represented by a set of link segments connecting each two adjacent vertices in the link.

In addition to geographical coordinates, a location in the road network can be referenced in a natural and convenient way using a linear reference system (LRS) (Miller and Shaw, 2001). The LRS has been widely used in transportation applications, such as recording traffic incidents and transportation infrastructures. A location, (x_i^u, y_i^u), on link a_u can be represented as (l_u, m_i), where $m_i \in (0,1)$ is the relative position on the link, and $m_i = 0.5$ indicates the middle of the link. The values $m_s = \varepsilon \approx 0$ and $m_e = 1-\varepsilon \approx 1$ respectively refer to the starting and ending nodes of the link, where ε is a very small tolerance (say $\varepsilon = 10^{-8}$). The linear reference (l_u, m_i) is an equivalent representation of the geographical location (x_i^u, y_i^u) in the road network (with the one-to-one relationship). The geographical location (x_i^u, y_i^u) can also be determined from (l_u, m_i) using the dynamic segmentation technique (Miller and Shaw, 2001).

Since linear measurement, m_i, is a real number between 0 and 1 and link ID, l_u, is a unique positive integer number, a compressed linear referencing (CLR) technique is proposed by integrating m_i and l_u into a single z_i^u value:

$$z_i^u = l_u + m_i \tag{18}$$

For example, control point c_d shown in Figure 6 locates on link 5 with $m_d = 0.45$, and this network location is referenced as $z_d^5 = 5.45$ using the CLR technique. It is apparent that the z_i^u value is also an equivalent representation of the geographical location (x_i^u, y_i^u) in the road network and can be easily transformed to the geographical location using the dynamic segmentation technique.

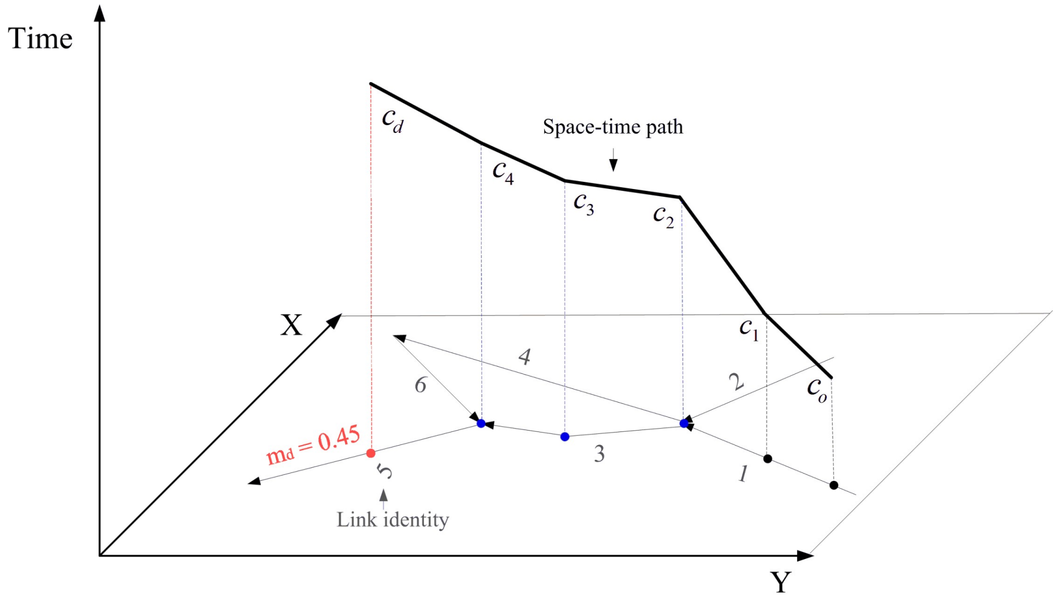


Figure 6. A network space-time path in (x, y, t) space

Using the CLR technique, any space-time location, $c_i^u =< x_i^u, y_i^u, t_i >$, on link l_u can be referenced in (z, t) space by

$$\&\%_i^u =< z_i^u, t_i > \qquad (19)$$

For example, the control point $c_d =< x_d, y_d, t_d >$ shown in Figure 6 can be represented as $\&\%_d^5 =< 5.45, t_d >$ in (z, t) space. In this way, the proposed CLR approach represents network time geographic entities as 2D elements in (z, t) space, which makes it possible to utilize existing 2D GIS platforms for storing and managing network time geographic entities. For convenience, (z, t) space is referred as CLR space in hereafter.

3.1. Network space-time path and station in CLR space

Figure 6 illustrates a network space-time path in the (x, y, z) space. Unlike planar space, movements in the road network are restricted by road geometry and traffic regulations (e.g. turn restrictions). To reflect road geometry in the space-time path, network nodes and intermediate vertices are required as control points. As shown in Figure 6, the network space-time paths should include not only observation locations (c_o, c_1 and c_d) but also network nodes (c_2 and c_4) and an intermediate vertex (c_3) as control points. In practice, observations are seldom collected at network nodes and intermediate vertices, and control points at these locations are generally estimated by linear interpolation (Equation (5)), assuming that the individual is moving at a constant speed between two adjacent observations.

Figure 7 illustrates the space-time path in CLR space using the same movement data shown in Figure 6. The space-time path $P\%$ is represented in CLR space by a set of disjoint LineString (i.e., polyline) elements

$$\tilde{P} = < \tilde{R}_u, \tilde{R}_v ..., \tilde{R}_w > \tag{20}$$

where the LineString element $\tilde{R}_u$ corresponds to a continuous movement on link a_u. For example, the space-time path shown in Figure 7 consists of three LineString elements ($\tilde{R}_1$, $\tilde{R}_3$, and $\tilde{R}_5$) representing movements on Links 1, 3, and 5 respectively. Although the space-time path is continuous in (x, y, t) space, the LineString elements are disjoint in CLR space.

Each LineString element, $\tilde{R}_u$, comprises a set of control points, $\{\tilde{c}^u_s ..., \tilde{c}^u_i, ..., \tilde{c}^u_e\}$, in CLR space, including $\tilde{c}^u_s$ at starting node n^u_s, $\tilde{c}^u_e$ at ending node n^u_e, and an arbitrary number of control points $\tilde{c}^u_i$ for LAT observations. All control points in $\tilde{R}_u$ are converted from corresponding control points in (x, y, t) space using the CLR technique. For example, $\tilde{c}^1_o$, $\tilde{c}^1_1$, and $\tilde{c}^1_e$ in Figure 7 are converted from c_o, c_1, and c_2 in Figure 6. Note that control points at intermediate vertices (e.g. c_3 in Figure 6) are no longer needed in $\tilde{R}_u$ to maintain road geometry. This can be important to reduce data volume for large-scale movement data.

In LineString element $\tilde{R}_u$, a straight line segment, $\tilde{s}_{ij} = < \tilde{c}^u_i, \tilde{c}^u_j >$, is connected between each two subsequent control points, $\tilde{c}^u_i = < z^u_i, t_i >$ and $\tilde{c}^u_j = < z^u_j, t_j >$. The speed of the line segment is

$$v^u_{ij} = (m_j - m_i) d_u / (t_j - t_i) \tag{21}$$

where m_j and m_i are decimal parts of z^u_j and z^u_i, respectively; and d_u is the link length. Thus, LineString element $\tilde{R}_u$ can be expressed as one or more straight line segments,

$$\tilde{R}_u = < ..., \tilde{s}_{ij}, ... > \tag{22}$$

The LineString element has at least one straight line segment between the starting node, $\tilde{c}^u_s$, and the ending node, $\tilde{c}^u_e$. For each line segment, $\tilde{s}_{ij}$, the z value at any time stamp $t_k \in [t_i, t_j]$ is determined by

$$\tilde{R}_u(t_k) = \{ z^u_k \mid z^u_k = (1-\lambda) z^u_i + \lambda z^u_j ; \lambda = (t_k - t_i)/(t_j - t_i) \} \tag{23}$$

Then, the location at any time stamp $t_k \in [t_o, t_d]$ in the space-time path $\tilde{P} = < \tilde{R}_u, \tilde{R}_v ..., \tilde{R}_w >$ can be determined from piecewise linear interpolation

$$\tilde{P}(t_k) = \{ z_k \mid z_k = \tilde{R}_u(t_k), t^u_s \le t_k \le t^u_e ; z_k = \tilde{R}_v(t_k), t^v_s < t_k \le t^v_e ; ... ; z = \tilde{R}_w(t_k), t^w_s < t_k \le t^w_e \} \tag{24}$$

We can prove that at any time stamp the location $z_k = \tilde{P}(t_k)$ is equivalent to the corresponding location $(x_k, y_k) = P^q(t_k)$ in (x, y, t) space as follows.

Proposition 1. The space-time path $\tilde{P}$ in CLR space is an equivalent representation of space-time path P^q in (x, y, t) space.

Proof. From the definition of $\tilde{P}^q$, all control points, $\{\tilde{c}_s^u ..., \tilde{c}_i^u, ..., \tilde{c}_e^u\}$, at network nodes and observations in CLR space are converted from the corresponding control points in P^q. Thus, at these control points, $z = \tilde{P}^q(t)$ is equivalent to $(x, y) = P^q(t)$. For any time stamp $t_k \in (t_i, t_j)$ between two control points, $\tilde{c}_i^u$ and $\tilde{c}_j^u$, $z_k^u = \tilde{P}^q(t_k)$ is determined by linear interpolation (Equation (22)). Since any space-time location between two observations is also estimated by linear interpolation in P^q, $z_k^u = \tilde{P}^q(t_k)$ is also equivalent to $(x_k, y_k) = P^q(t_k)$. □

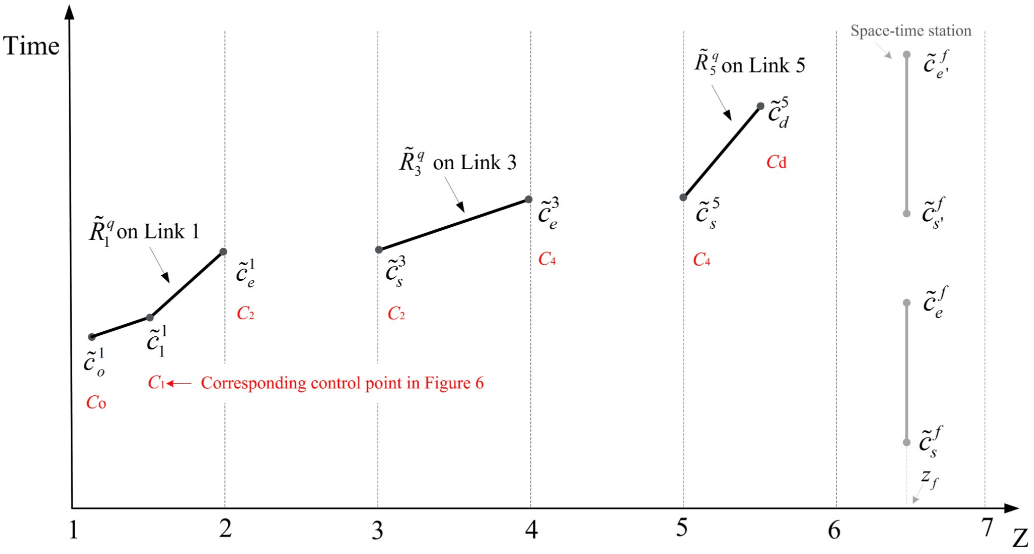


Figure 7. A network space-time path in the CLR space

Figure 7 also illustrates the space-time station in CLR space (denoted by $\tilde{Q}^f$). The space-time station is represented in CLR space by a set of disjoint vertical LineString elements

$$\tilde{Q}^f = < \tilde{R}_{se}^f, \tilde{R}_{s'e'}^f, ... > \tag{25}$$

Each LineString element, $\tilde{R}_{se}^f$, refers to an available time duration, $[t_s^f, t_e^f]$, at activity location, z_f. It can be expressed as a vertical line segment, $\tilde{R}_{se}^f = < \bar{s}_{se} >$, between two control points, $\tilde{c}_s^f = < z_f, t_s^f >$ and $\tilde{c}_e^f = < z_f, t_e^f >$. The space-time station is stationary for the available time durations as

$$\tilde{Q}^f(t_k) = \{z_k \mid z_k = z_f, \forall t_k \in ([t_s^f, t_e^f], [t_{s'}^f, t_{e'}^f], ...)\} \tag{26}$$

Lemma 1. The space-time station $\tilde{Q}^f$ in CLR space is an equivalent representation of space-time station Q^f in (x, y, t) space.

Proof. It is apparent that $z_f = \tilde{Q}^f(t_k)$ is equivalent to $(x_f, y_f) = Q^f(t_k)$, since it is stationary during all available time periods. □

3.2. Network space-time prism and lifeline in CLR space

Space-time prisms in the road network are restricted by the road geometry. In the road network, the shortest distance between any two locations (x_1, y_1) and (x_2, y_2) are not Euclidean distance but the network distance, denoted by $\tilde{D}(x_1, y_1, x_2, y_2)$, which can be calculated by the shortest path algorithms (Li et al., 2015). To represent the essential ideas of network space-time prisms, it is assumed that U-turn is not allowed along any network link. This assumption is reasonable for major divided roads. For minor undivided roads, such assumption can be relaxed according to Kuijpers and Othman (2009).

Figure 8 illustrates the network space-time prism in (x, y, t) space using $c_o =< x_o, y_o, t_o >$ and $c_d =< x_d, y_d, t_d >$ as anchor points. The network space-time prism is a 3D element in (x, y, t) space delimiting all accessible space-time locations in the road network. Since network links are 1D linear features, the network space-time prism comprises a set of 2D polygons; each of which represents the potential space-time locations on a corresponding network link. Let O_{od}^u denote the 2D space-time polygon in (x, y, t) space on link a_u. The network space-time prism can be represented as a set of space-time polygons in (x, y, t) space

$$STP_{od} =< O_{od}^u, ... > \tag{27}$$

For example, the network space-time prism shown in Figure 8 consists of 5 polygons in space and time (O_{od}^1, O_{od}^3, O_{od}^4, O_{od}^5, and O_{od}^6).

Each space-time polygon, O_{od}^u, comprises a set of control points $\{c_s^-, c_i^- ..., c_e^-, c_e^+, ..., c_i^+, c_s^+\}$ at starting node n_s^u, ending node n_e^u, and all intermediate vertices, n_i^u. For each node $n_i^u =< x_i^u, y_i^u >$ (or vertex), two controls, $c_i^- =< x_i^u, y_i^u, t_i^- >$ and $c_i^+ =< x_i^u, y_i^u, t_i^+ >$, are constructed at its earliest arrival time, t_i^- , and latest departure time, t_i^+ , respectively. The space-time polygon, O_{od}^u, is represented by a ring connecting all control points in clockwise order:

$$O_{od}^u =< s_{si}^{--}, ..., s_{ee}^{-+}, ..., s_{ss}^{+-} > \tag{28}$$

where s_{si}^{--} is the line segment connecting two control points c_s^- and c_i^-.

Solution algorithms have been proposed in the literature to generate the network space-time prism by explicitly constructing space-time polygons based on forward and backward path searches from c_o and c_d, respectively (Chen et al., 2013; Miller, 1991). If two anchor points, c_o and c_d, are not at network nodes, two dummy nodes, $n_o = (x_o, y_o)$ and $n_d = (x_d, y_d)$, would be added to the road network. The forward shortest path search determines the least travel time $\tilde{T}(n_o, n_s^u)$ from the starting anchor point, $n_o = (x_o, y_o)$, to each network node, $n_s^u =< x_s^u, y_s^u >$, while the backward search determines the least travel time $\tilde{T}(n_s^u, n_d)$ from each network node, $n_s^u =< x_s^u, y_s^u >$, to the ending anchor point, $n_d = (x_d, y_d)$. The accessible link a_u can be determined from constraints $\tilde{T}(n_o, n_s^u) + \tilde{T}(n_s^u, n_d) \le t_d - t_o - c_{\min}$ and $\tilde{T}(n_o, n_e^u) + \tilde{T}(n_e^u, n_d) \le t_d - t_o - c_{\min}$. For starting node n_s^u on each link a_u, the earliest arrival time, $t_s^- = t_o + \tilde{T}(n_s, n_s^u)$, and the latest departure time, $t_s^+ = t_d - \tilde{T}(n_s^u, n_e)$, are calculated to construct control points $c_s^- =< x_s^u, y_s^u, t_s^- >$ and $c_s^+ =< x_s^u, y_s^u, t_s^+ >$. The earliest arrival time, $t_e^- = t_s^- + \tau_u$, and latest departure time, $t_e^+ = t_s^+ + \tau_u$, at the ending node, n_e^u, are then obtained to construct control points $c_e^- =< x_e^u, y_e^u, t_e^- >$ and $c_e^+ =< x_e^u, y_e^u, t_e^+ >$. The earliest arrival time, $t_i^- = t_s^- + m_i^u \tau_u$, and latest departure time, $t_i^+ = t_s^+ + m_i^u \tau_u$, at every intermediate vertex, n_i^u, are obtained to determine $c_i^- =< x_i^u, y_i^u, t_i^- >$ and $c_e^+ =< x_i^u, y_i^u, t_i^+ >$, where m_i^u is the relative position of the vertex along link a_u. After all control points $\{c_s^-, c_i^- ..., c_e^-, c_e^+, ..., c_i^+, c_s^+\}$ are constructed, the space-time polygon, O_{od}^u, is generated for each accessible link a_u.

As shown in Figure 8, the projection of the network space-time prism onto geographical space is the potential network area (PNA) in terms of accessible links. It can be defined as those links with starting and ending nodes satisfying time constraints:

$$PNA_{od} = \left\{ a_u \mid \left(t_s^+ - t_s^- \ge c_{\min} \right) \wedge \left(t_e^+ - t_e^- \ge c_{\min} \right) \right\} \tag{29}$$

Given any location (x_k^u, y_k^u) on link a_u, its earliest arrival time t_k^- and latest departure time t_k^+ can be determined by

$$t_k^- = t_s^- + m_k^u \tau_u ; t_k^+ = t_s^+ + m_k^u \tau_u \tag{30}$$

where m_k^u is the relative position of location (x_k^u, y_k^u) along link a_u. The potential locations of the individual at a given time instance $t_k \in [t_o, t_d]$ are a subset of PNA,

$$STP_{od}(t_k) = \left\{ (x_k, y_k) \mid \left(t_k^- \le t \right) \wedge \left(t_k^+ \ge t \right) \wedge \left(t_k^+ - t_k^- \ge c_{\min} \right) \right\} \tag{31}$$

The height of the space-time prism at any location (x_k^u, y_k^u) on link a_u is determined by its earliest arrival time t_k^- and latest departure time t_k^+

$$STP_{od}(x_k, y_k) = \{ t_k \mid t_k^- \le t_k \le t_k^+ \} . \tag{32}$$

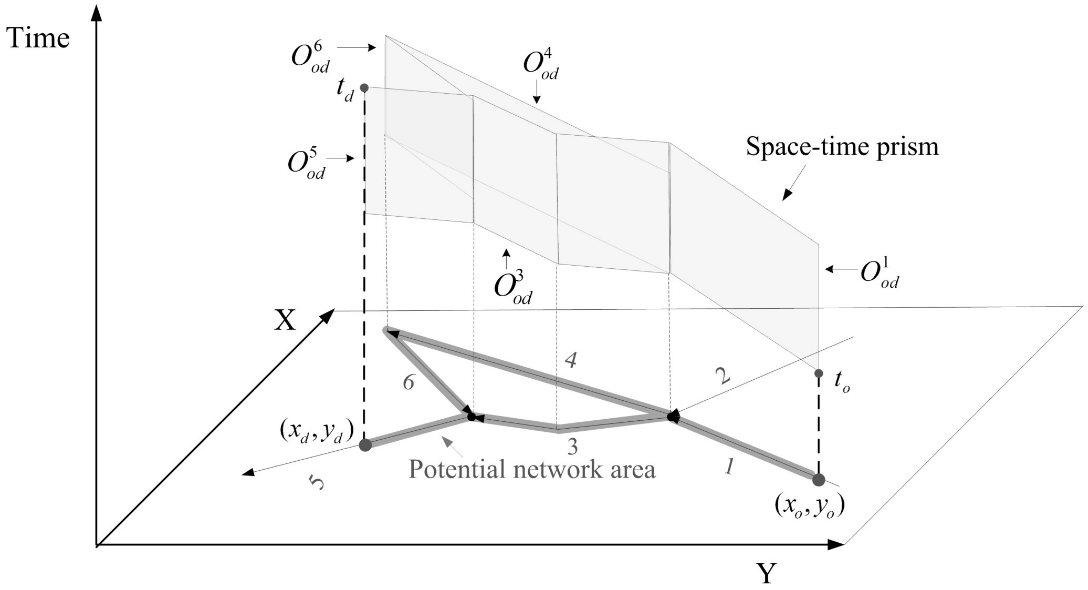


Figure 8. A network space-time prism in (x, y, t) space

Figure 9 illustrates the network space-time prism in CLR space using the same settings as Figure 8. The two anchor points are $\tilde{c}_o =< z_o, t_o >$ and $\tilde{c}_d =< z_d, t_d >$ in CLR space. As shown in the figure, the network space-time prism in CLR space ($\tilde{S}$TP$_{od}$) comprises a set of 2D polygons:

$$\tilde{S}TP_{od} =< \tilde{O}_{od}^{u}, \ldots >, \forall a_u \in PNA \tag{33}$$

where each polygon, $\tilde{O}_{od}^{u}$, corresponding to a space-time polygon, O_{od}^{u}, in (x, y, t) space, delimits all potential space-time locations on the link a_u. For example, the network space-time prism shown in Figure 9 consists of five polygons in CLR space ($\tilde{O}_{od}^{1}$, $\tilde{O}_{od}^{3}$, $\tilde{O}_{od}^{4}$, $\tilde{O}_{od}^{5}$, and $\tilde{O}_{od}^{6}$) corresponding to the five space-time polygons in Figure 8.

Each polygon $\tilde{O}_{od}^{u}$ consists of only four control points $\{\tilde{c}_s^{-}, \tilde{c}_e^{-}, \tilde{c}_e^{+}, \tilde{c}_s^{+}\}$ in CLR space. Two control points $\tilde{c}_s^{-} =< z_s^u, t_s^- >$ and $\tilde{c}_s^{+} =< z_s^u, t_s^+ >$ refer to the earliest arrival time, t_s^-, and the latest departure time, t_s^+, at the starting node, z_s^u, respectively. The other two control points, $\tilde{c}_e^{-} =< z_e^u, t_e^- >$ and $\tilde{c}_e^{+} =< z_e^u, t_e^+ >$, correspond to the earliest arrival time, t_e^-, and the latest departure time, t_e^+, at the ending node, z_e^u. All four control points can be transformed from the corresponding control points in O_{od}^{u} in (x, y, t) space using the CLR technique. Control points at intermediate vertices are no longer needed in $\tilde{O}_{od}^{u}$ to represent road geometry. Thus the space-time polygon, $\tilde{O}_{od}^{u}$, can be represented by a ring connecting four control points in clockwise order:

$$\tilde{O}_{od}^{u} =< \tilde{s}_{se}^{--}, \tilde{s}_{ee}^{-+}, \tilde{s}_{es}^{++}, \tilde{s}_{ss}^{+-} > \tag{34}$$

where $\tilde{s}_{se}^{--}$ is the line segment connecting two control points, $\tilde{c}_s^{-}$ and $\tilde{c}_e^{-}$.

The network space-time prism in CLR space, $\tilde{\mathrm{S}}\mathrm{TP}_{\mathrm{od}}$, can be constructed by existing solution algorithms based on forward and backward shortest path searches. Using the results of shortest path searches, the earliest arrival time, $t_s^- = t_o + \tilde{T}(n_o, n_s^u)$, and the latest departure time, $t_s^+ = t_d - \tilde{T}(n_s^u, n_d)$, can be calculated, to construct $\tilde{c}_s^- =< z_s^u, t_s^- >$ and $\tilde{c}_s^+ =< z_s^u, t_s^+ >$ at starting node n_s^u on each accessible link a_u. The earliest arrival time, $t_e^- = t_s^- + \tau_u$, and latest departure time, $t_e^+ = t_s^+ + \tau_u$, at ending node n_e^u can then be calculated, to construct control points $\tilde{c}_e^- =< z_e^u, t_e^- >$ and $\tilde{c}_e^+ =< z_e^u, t_e^+ >$. After constructing these four control points $\{\tilde{c}_s^-, \tilde{c}_e^-, \tilde{c}_e^+, \tilde{c}_s^+\}$, the space-time polygon, $\tilde{O}_{od}^u$, on accessible link a_u is generated. Consequently, the space-time prism, $\tilde{S}TP_{od} =< \tilde{O}_{od}^u, ... >$, is constructed by generating space-time polygons for all accessible links.

The projection of the space-time prism onto the z axis is the PNA in terms of linear events $\{(z_s^u, z_e^u), ...\}$. Each linear event, (z_s^u, z_e^u), represents an accessible link, a_u, in CLR space. Thus, the PNA in CLR space ($\tilde{P}NA_{od}$) can be expressed as

$$\tilde{P}NA_{od} = \left\{ (z_s^u, z_e^u) \mid \left(t_s^+ - t_s^- \geq c_{\min} \right) \wedge \left(t_e^+ - t_e^- \geq c_{\min} \right) \right\} \tag{35}$$

Given at any location, z_k^u, on link a_u, its earliest arrival time, t_k^-, and latest departure time, t_k^+, can be determined by

$$t_k^- = t_s^- + m_k^u \tau_u ; t_k^+ = t_s^+ + m_k^u \tau_u \tag{36}$$

where m_k^u is the decimal part of z_k^u. The potential locations of the individual at a given time stamp, $t_k \in [t_o, t_d]$, are a subset of PNA:

$$\tilde{S}TP_{od}(t_k) = \left\{ z_k \mid \left(t_k^- \leq t \right) \wedge \left(t_k^+ \geq t \right) \wedge \left(t_k^+ - t_k^- \geq c_{\min} \right) \right\} \tag{37}$$

We can prove that the potential locations in $\tilde{S}TP_{od}(t_k)$ at any time stamp $t_k \in [t_o, t_d]$ are equivalent to corresponding locations in $\mathrm{STP}_{\mathrm{od}}(\mathrm{t}_{\mathrm{k}})$ in (x, y, t) space as follows.

Proposition 2. The network space-time prism $\tilde{\mathrm{S}}\mathrm{TP}_{\mathrm{od}}$ in CLR space is an equivalent representation of the network space-time prism $\mathrm{STP}_{\mathrm{od}}$ in (x, y, t) space.

Proof. From the definition of $\tilde{\mathrm{S}}\mathrm{TP}_{\mathrm{od}}$, four control points, $\{\tilde{c}_s^-, \tilde{c}_e^-, \tilde{c}_e^+, \tilde{c}_s^+\}$, at network nodes of each space-time polygon $\tilde{O}_{od}^u$ are converted from corresponding control points, $\{c_s^-, c_e^-, c_e^+, c_s^+\}$, in O_{od}^u using the CLR technique. Therefore, the earliest arrival time, t_s^-, and the latest departure time, t_s^+, at the starting node in $\tilde{O}_{od}^u$ are identical to that in O_{od}^u. Comparing Equations (36) and (30), we have the earliest arrival time, t_k^-, and the latest departure time, t_k^+, at any location z_k^u in CLR space identical to that at corresponding location (x_k^u, y_k^u) in (x, y, t) space. Since t_k^- and t_k^+ are identical at any location in the network, the potential locations in $\tilde{STP}_{od}(t_k)$ are therefore equivalent to $\mathrm{STP}_{\mathrm{od}}(\mathrm{t_k})$ from Equations (37) and (31). □

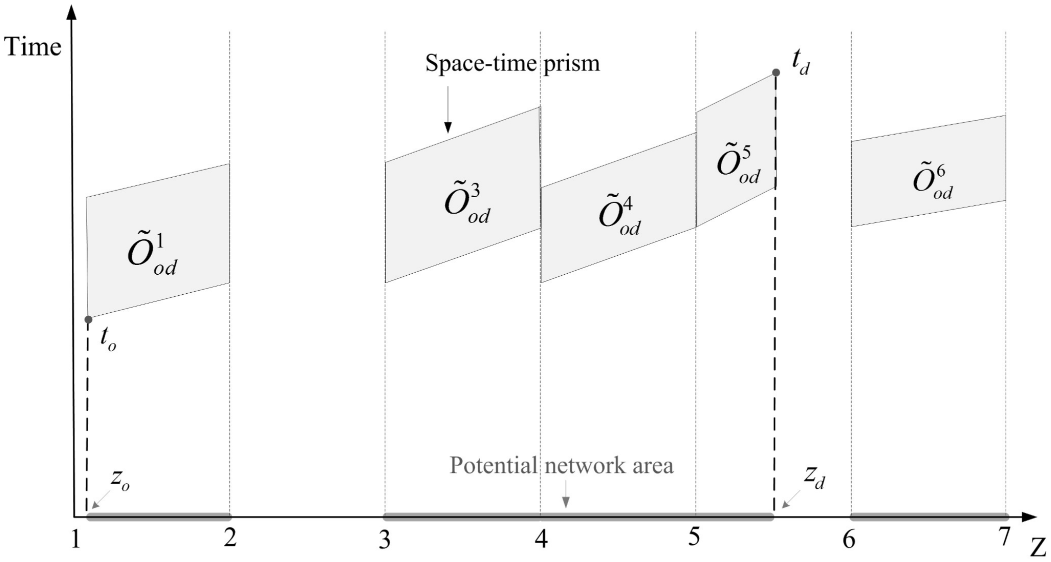


Figure 9. A network space-time prism in the CLR space

Figure 10 illustrates a simple network space-time lifeline in CLR space. The illustrated space-time lifeline is a combination of a space-time path and two network space-time prisms in CLR space. The network space-time prism is a 2D entity while the space-time path is 1D (linear) entity. Since it is difficult to represent different dimensional entities in a feature, every space-time path in the lifeline is converted into 2D entity using spatiotemporal buffering. Given a space-time path $\tilde{P} =< \tilde{R}_u, \tilde{R}_v ..., \tilde{R}_w >$ and a spatial tolerance ω, this operation creates a spatial buffering on the individual's location, $\tilde{P}(t)$, any time stamp, t. An adoptive spatial tolerance $\omega = \eta d_u$ can also be used for links with different lengths, where d_u is the link length and η is threshold value in percentage. The result of spatiotemporal buffering (space-time buffer, $\tilde{B}$) is the space-time regions within which the distance of any location to $\tilde{P}(t)$ is less than tolerance ω as

$$\tilde{B}(t) = \{z \mid \tilde{D}\left(z, \tilde{P}(t)\right) \le \omega\} \tag{38}$$

where $\tilde{D}\left(z, \tilde{P}(t)\right)$ is the network distance from location z to location $\tilde{P}(t)$. The space-time buffer consists of a set of space-time polygons, $\tilde{B} =< \tilde{O}_u, \tilde{O}_v ..., \tilde{O}_w >$, and each $\tilde{O}_u$ represents the space-time regions on link a_u. The detailed algorithm for spatiotemporal buffering operation in CLR space is described in Chen et al. (2015).

After all space-time paths are converted into space-time buffers using a very small spatial tolerance, $\omega \approx 0$, the space-time lifeline in CLR space, $\tilde{L}$, can be represented by a set of 2D space-time polygons from both space-time buffers and prisms:

$$\tilde{L} = \{\tilde{O}_u, \ldots; \tilde{O}_{od}, \ldots\} \tag{39}$$

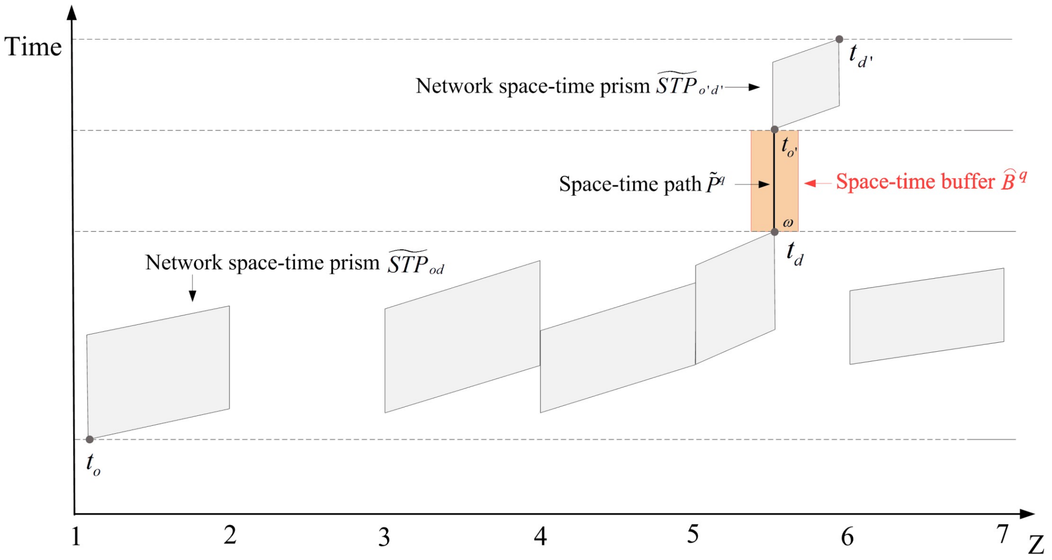

Time
$t_{d'}$
Network space-time prism $\widetilde{STP}_{o'd'}$
$t_{o'}$
Space-time path $\tilde{P}^q$
Space-time buffer $\widehat{B}^q$
ω
t_d
Network space-time prism $\widetilde{STP}_{od}$
t_o
1
2
3
4
5
6
7
Z

Figure 10. A network space-time lifeline in the CLR space

Lemma 2. The space-time lifeline, $\tilde{L}$, in CLR space is an equivalent representation of space-time lifeline, L^w, in (x, y, t) space.

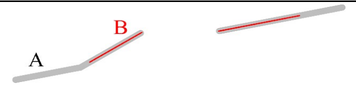
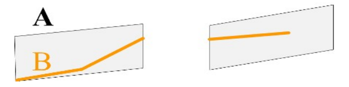
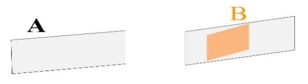
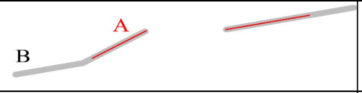
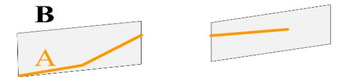
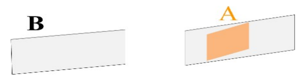
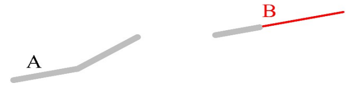
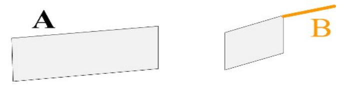
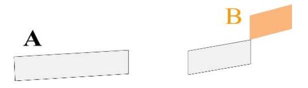
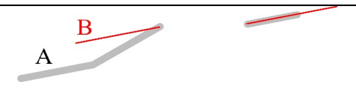
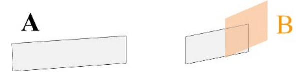
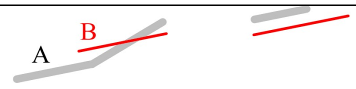
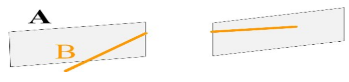
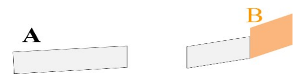
Proof. The space-time lifeline, L^w, in (x, y, t) space is a combination of space-time paths and prisms. From Proposition 2, all space-time prisms in $\tilde{L}$ in CLR space are equivalent representations of corresponding prisms in (x, y, t) space. From the definition of $\tilde{L}$, all space-time paths are converted to space-time buffers by spatiotemporal buffering using a very small spatial tolerance. Thus, space-time buffers in $\tilde{L}$ can approximate the space-time paths in CLR space. From Proposition 1, these space-time buffers are equivalent representations of corresponding space-time paths in (x, y, t) space. Therefore, the space-time lifeline, $\tilde{L}$, is an equivalent representation of L^w in (x, y, t) space. □

3.3. Network time geographic relations in CLR space

As noted above, space-time intersection and path bundling are two key space-time relationships in time geography. Table 1 shows the space-time intersections for network time geographic entities in CLR space. Since the space-time paths (and stations) are 1D entities and the space-time prisms (and lifelines) are 2D entities in CLR space, space-time intersections can be classified into path-path, path-prism and prism-prism cases. All these space-time intersection problems in CLR space can be equivalently solved by classical spatial intersection in 2D space (Egenhofer and Herring, 1990).

As shown in Table 1, the space-time intersections in this study are further refined into five space-time sub-relations: *Contains*, *Within*, *Touches*, *Overlaps* and *Crosses*. Contains is the condition that Entity A wholly contains Entity B. For example, space-time path B shown in the table is a feasible schedule for an individual, because the individual's space-time prism B contains the path. Within is the complementary condition that Entity A is wholly contained within Entity B. Touches refers to the condition of Entity A's ending point meeting Entity B's starting point. For example, an individual's space-time prism A in the table touches his space-time path B indicating that a fixed activity (i.e. path) would be carried out after a flexible activity (i.e. prism). Overlaps is a space-time relation of same dimension entities (i.e. path-path and prism-prism), and refers to the coincidence of Entities A and B for a time period. For example, a joint activity can be scheduled only at space-time locations where space-time prism A overlaps prism B, as shown in the table. Crosses is valid for all path-path, path-prism and prism-prism cases. It refers to where the intersection of the two entities has a lower dimension than the maximum dimension of the entities. For the path-path case, Crosses refers to the coincidence of Entities A and B at certain time instances except for the Touches conditions. For the path-prism case, Crosses is the condition of Entity A intersecting Entity B for certain time periods or time instances except for the Touches and Contains conditions. For the prism-prism case, the Crosses is the condition that the intersection of two prisms is one or more 1D space-time paths.

Table 1. Space-time intersections of network time geographic entities in the CLR space

Relations	Path-Path	Path-Prism	Prism-Prism
A Contains B			
A Within B			
A Touches B			
A Overlaps B		-	
A Crosses B			

Path bundles correspond to the convergence of space-time paths in space and time. Bundling very often occurs at some stations, implying that the individuals are participating in shared activities at the stations. Paths can also bundle during the movements, such as taking the same bus. Unlike space-time intersections, bundling of two space-time paths P^q and P^r for a time period, $[t_{B'}, t_{B''}]$, does not require exact co-existence of these two paths in space but allows a spatial tolerance, ω. Thus, these two paths should satisfy $\tilde{D}\left(P^q(t), P^r(t)\right) \le \omega$ for any time instance $\forall t \in [t_{B'}, t_{B''}]$, where $\tilde{D}(,)$ is the network distance between two locations.

As shown in Figure 11, using spatiotemporal buffering, path bundling can be solved as a space-time intersection problem. Given a spatial tolerance, ω, a space-time buffer, $\bar{B}^f$, can be generated for a space-time station, $\bar{Q}^f$, during the time period $[t_{B'}, t_{B''}]$. The generated space-time buffer $\bar{B}^f$ delimits all space-time locations whose network distance to station $\bar{Q}^f(t)$ at any time instance, $\forall t \in [t_{B'}, t_{B''}]$, is less than the spatial tolerance. Therefore, all space-time paths bundling at the station $\bar{Q}^f$ can be determined by the Contains relation as those paths contained by the space-time buffer $\bar{B}^f$ during the time period $[t_{B'}, t_{B''}]$. Similarly, paths bundling during the movements can also be determined by generating space-time buffer $\bar{B}^q$ for the movement $P^q(t)$ during the time period $[t_{B'}, t_{B''}]$. The paths bundling with the target path P^q are those paths contained by the space-time buffer, $\bar{B}^q$, during the time period $[t_{B'}, t_{B''}]$ (e.g. P^r in Figure 11). The detailed algorithm for generating space-time buffers in CLR space is given in Chen et al. (2015).

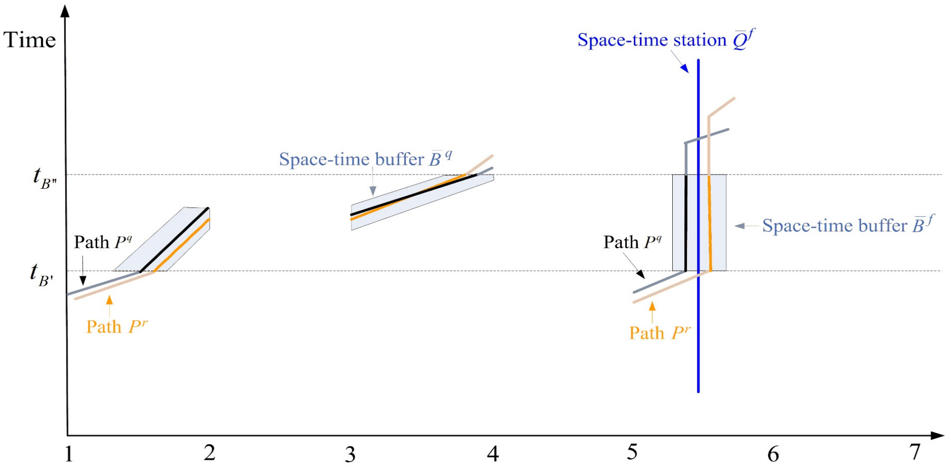

Time
$t_{B''}$
$t_{B'}$
Space-time station $\bar{Q}^f$
Space-time buffer $\bar{B}^q$
Path P^q
Path P^r
Path P^q
Space-time buffer $\bar{B}^f$
Path P^r
1
2
3
4
5
6
7

Figure 11. Paths bundling in the CLR space

4. Spatiotemporal data model for network time geographic entities in CLR space

This section presents the spatiotemporal data model, associated spatiotemporal database design and spatiotemporal queries based on the above representations of network time geographic entities and relations in CLR space.

4.1. Spatiotemporal data model

Figure 12 shows the class diagram of the spatiotemporal data model for network time geographic entities in CLR space. To integrate with existing GIS platforms, the OpenGIS simple features access standard (Herring, 2011) is adopted to represent classical 2D spatial features (i.e., Geometry, Point, MultiPoint, LineString, MultiLineString, Polygon, and MultiPolygon). As shown in the figure, ClrSpaceTimePath, ClrSpaceTimeStation, ClrSpaceTimePrism, ClrSpaceTimeLifeLine, and ClrSpaceTimeBuffer represent the space-time path, station, prism, lifeline, and buffer entities, respectively, in CLR space. A

ClrObservations object is also introduced to represent discrete LAT observations. These space-time entities are all derived from the Geometry object, implying that they are special types of 2D spatial objects in CLR space. The space-time path and station entities are represented by a MultiLineString object comprising a set of LineString objects. The space-time prism, lifeline, and buffer entities are formulated by a MultiPolygon object consisting of one or more Polygon objects.

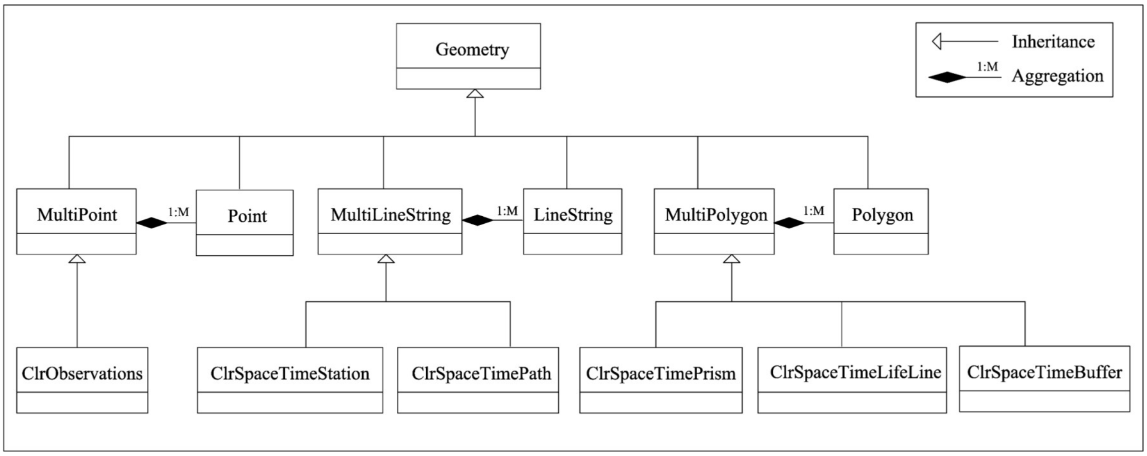


Figure 12. Spatiotemporal data model for network time geographic entities in CLR space

4.2. Spatiotemporal database design

Based on the above spatiotemporal data model, the network time geographic entities can be stored and managed in classical spatial databases, such as ESRI Geodatabase or Oracle Spatial. Thus, space-time entities with the same dimension can be organized and stored as a feature class (or a layer). For example, space-time prisms of a collection of individuals are organized as a polygon feature class and each record of the feature class is the space-time prism of an individual. In addition to the spatiotemporal shape, semantic information can be stored as attributes in the feature class, such as individuals' age, sex, occupation, etc. Since spatial database techniques are adopted, traditional spatial datasets, including road networks, can be stored and managed in the same database. This makes it easy to integrate time geographic analysis with the powerful visualization and spatial analysis capabilities provided by contemporary GIS platforms (Kwan, 2000).

Since the network time geographic entities are modelled as 2D features in CLR space, the numerous spatial operations developed for contemporary GIS platforms can be directly utilized to implement the spatiotemporal operations of network time geographic entities. For example, the spatial relational operators (i.e. IRelationalOperator) provided in ArcGIS can be directly employed to determine if a certain space-time intersection relationship defined in Table 1 exists between two time geographic entities. The Intersect function of ITopologicalOperator in ArcGIS can be utilized to obtain the intersection result of two time geographic entities. The use of well-developed spatial operations provides an effective approach for quickly developing time geographic analysis applications. Importantly, this approach can lead to more efficient time geographic analysis, because it is always computationally simpler to manipulate entities in 2D CLR space than the higher dimensional (x, y, t) space.

The well-developed spatial index technique can also be directly employed to construct the spatiotemporal index for quick retrieval of network time geographic entities in the database. Many existing spatial index structures from the literature can be used, including R-tree (Guttman, 1984), Quad-tree (Finkel and Bentley, 1974), and Grid-file (Nievergelt et al., 1984).

4.3. Spatiotemporal queries

Based on the spatial database and index techniques, spatiotemporal queries on time geographic entities can be easily implemented using the spatial query technique. Similar to traditional spatial queries, one requires an input space-time geometry and to choose a space-time relation defined in Table 1 (or roughly the intersection relation representing any space-time intersection relation). Using different values of these two parameters, various spatiotemporal queries can be performed to retrieve the time geographic entities stored in the database. Some typical spatiotemporal queries are as follows.

1. *Path-path intersection query. Given a space-time path, $\tilde{P}$, of an individual, find all space-time paths intersecting the path.*
   This spatiotemporal query retrieves all individuals who physically meet the given individual. The query can be easily implemented in CLR space using a spatial intersect query on a space-time path layer with $\tilde{P}$ as the input geometry.

2. *Window-path intersection query. Given a spatial window, $W = (x_1, x_2, y_1, y_2)$, and a time period, $[t_o, t_d]$, find all space-time paths intersecting the spatial region during the time period t_o to t_d.*
   This spatiotemporal query finds all individuals who exist in the spatial region W during the time period from t_o to t_d. The query can be implemented in four steps. (1) A spatial query is performed on the road network to retrieve all network links $a_u,...,a_v$ intersecting the input spatial window W. (2) The a set of linear events, $\{[z_i^u, z_j^u],...,[z_i^v, z_j^v]\}$, is determined within the spatial window for all retrieved networks. (3) A space-time region, $\tilde{W}$, in CLR space is constructed by generating a set of space-time polygons, $\tilde{O},...,\tilde{O}$. Each space-time polygon $\tilde{O}$ is determined by four control points $\tilde{p}_i =< z_i^u, t_o >$, $\tilde{p}_i =< z_i^u, t_d >$, $\tilde{p}_j =< z_j^u, t_o >$, and $\tilde{p}_j =< z_j^u, t_d >$ on a linear event, $[z_i^u, z_j^u]$, during the

time period $[t_o, t_d]$. (4) A spatial intersect query is performed on a space-time paths layer using $\overrightarrow{W}$ as the input geometry.

3. *Station-prism within query. Given a location,* z_i^u*, and a time period,* $[t_o, t_d]$*, find all space-time prisms containing the location* z_i^u *for the time period* $[t_o, t_d]$*.*
   This spatiotemporal query finds all individuals who could feasibly conduct an activity at given location, z_i^u, during the time period from t_o to t_d. The query can be easily implemented in CLR space using a spatial Within query on a space-time prism layer. The input geometry is a space-time station at z_i^u for the time period $[t_o, t_d]$.

4. *Prism-prism intersection query. Given a space-time prism,* $\overrightarrow{STP}_{od}$*, of an individual, find all space-time prisms that intersect with the given prism* $\overrightarrow{STP}_{od}$*.*
   This spatiotemporal query finds all individuals who are feasible to conduct a joint activity with the given individual. The query can be easily implemented in CLR space using a spatial Intersect query on a space-time prism layer with $\overrightarrow{STP}_{od}$ as the input geometry.

5. Experimental evaluation

This section presents a case study using a real-world large-scale LAT observation data to demonstrate the applicability of the proposed spatiotemporal data model. In this study, the ArcGIS Geodatabase data model, which follows the OpenGIS simple features access standard, was used to implement the proposed spatiotemporal data model. ArcSDE 10.2 and Oracle 11g were adopted as the spatial database, which utilize the Grid-file index structure as the spatial index. In the database, Double data type was used for storing x, y, t and z elements. Spatiotemporal queries were implemented in the ArcEngine 10.2 development kit using Microsoft Visual C#. All experiments were conducted on a desktop computer with an Intel dual-core 3.1 GHz CPU, 8 GB RAM, running the Windows 7 64-bit operating system.

5.1. Data description

Three real-world datasets in Shenzhen city, China were collected for this case study. The first dataset was the Shenzhen road network, as shown in Figure 13, which consists of 69,198 nodes and 192,582 directed links. The identities of the network links are unique and range from 1 to 192,582. The road network was stored in the database as a road layer and a R-tree index was built for quickly retrieving the network links. The road network serves as a reference framework in CLR space for converting locations in (x, y, t) space to CLR space and vice versa. The distance (or travel time) between any two locations in CLR space can be calculated by the shortest path algorithm in the road network (Huang et al., 2007; Li et al., 2015). To facilitate these calculations, the road network was loaded into memory.

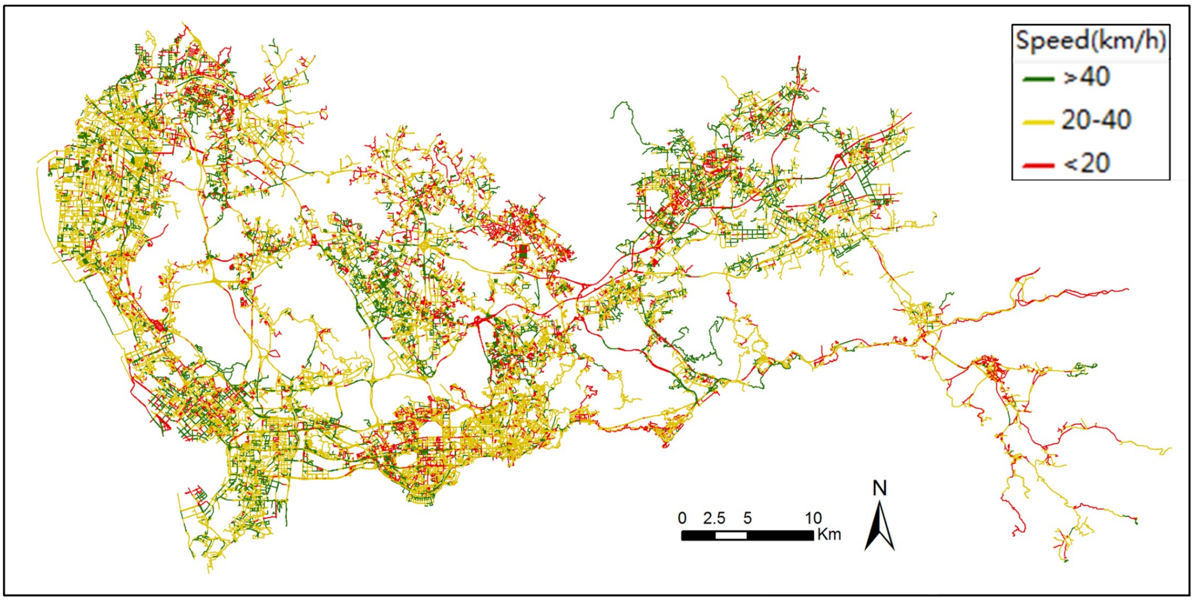


Figure 13. Shenzhen road network

The second dataset was Shenzhen floating car data. The floating car system in Shenzhen consists of ten thousands of taxis equipped with GPS devices to record their locations approximately every thirty seconds. Due to GPS positioning errors and the abstract representation of the road network, GPS observations of a taxi may not be exactly on network links. To address this, a multi-criteria dynamic programming map matching (MDP-MM) algorithm (Chen et al. 2014) was employed to accurately match the observations onto the road network. Since a relatively low frequency was used in the collected dataset, a taxi may have passed through several network links between two adjacent observations. The missing network nodes in the taxi trajectories can be imputed by the MDP-MM algorithm. After the map-matching process, all taxi trajectories were converted into CLR space and stored as a space-time path layer. The conversion of space-time paths from (x, y, t) space to CLR space did not introduce additional computational effort, because the z value of each GPS observation was already determined in the map matching process. Consequently, the trajectories of 13,445 taxis collected on a typical weekday (11 May 2012, a Friday) during a one hour period (8–9 a.m.) were converted into space-time paths in CLR space and stored in the spatial database as a polyline layer.

The collected taxi space-time paths were also utilized to estimate the traffic conditions of the Shenzhen road network. As discussed above, the space-time path, $\overline{P}^q$, of a taxi, q, records the time stamp of the taxi entering and exiting each link along the path. The travel time, τ_u^q, of the taxi on link a_u can be determined by the time difference between the taxi entering and exiting the link a_u. By calculating the travel times for all taxis passing the same link a_u, the travel time τ_u can be estimated by the corresponding mean value. Figure 13 shows the estimated traffic conditions of the Shenzhen road network. In the figure, red represents highly congested links (< 20 km/h); yellow slightly congested links (20–40 km/h); and green uncongested links (> 40 km/h).

The third dataset was mobile phone tracking data in Shenzhen city. This dataset was obtained from a telecommunication company and comprised 10,000 mobile phone users. The dataset was collected for the same day and time period as the floating car dataset. The locations of each mobile user were referenced by the user's nearest mobile phone tower, and the average positioning accuracy in the urban area was approximately 300 meters. The sampling frequency was one hour, so each mobile phone user had two observations at 8 and 9 a.m. Based on the traffic conditions estimated from the floating car data, a space-time prism in CLR space (i.e., $\tilde{S}TP_{od}$) was constructed for each mobile phone user. The minimum activity duration, $c_{\min}$, was set as 30 minutes for all mobile phone users. All the constructed space-time prisms were stored in the spatial database as a polygon layer.

5.2. Storage performance

The storage performance of the proposed spatiotemporal data model in CLR space was examined by comparison to the classical data model in (x, y, t) space.

Figure 14 shows the storage space required for the Shenzhen floating car dataset in CLR space. Five datasets were extracted from the original floating car dataset with different numbers of taxis. The storage space required by the proposed spatiotemporal data model in CLR space increased linearly with the number of taxis. For example, 6.83 MB was required for storing the space-time paths of 1,000 taxis, which increased by 13.8 times to 94.35 MB when the number of taxis increased to 13,445.

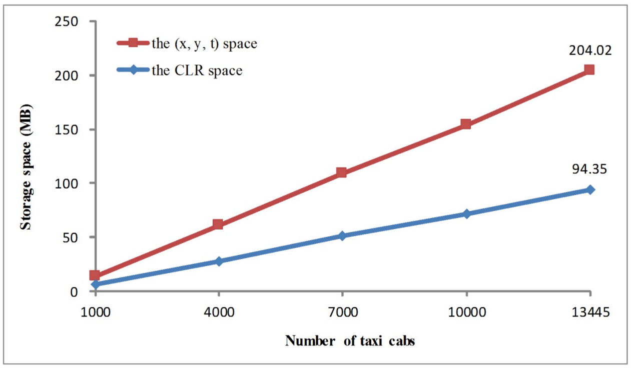


Figure 14. Storage space of space-time paths

Figure 14 also shows the storage space required for storing the same floating car dataset in (x, y, t) space. It can be observed from the figure that the proposed spatiotemporal data model has a significant storage advantage over the classical data model. For example, the storage space required for storing space-time paths of 13,445 taxis in CLR space is approximately 46.2% of that required for storing the same dataset in (x, y, t) space. This result is expected for two reasons. Firstly, the proposed spatiotemporal data model significantly reduces the number of control points required to store the space-time paths. To store space-time paths in classical (x, y, t) space, three types of control points are needed: GPS observations, network nodes, and link intermediate vertices. For example, 8,913,940 control points were used in (x, y, t) space to store 13,445 taxi space-time paths; and approximately 31% of these were link intermediate vertices. Using the proposed spatiotemporal data model in CLR space, the link intermediate vertex control points are no longer needed to maintain road geometry. Secondly, each control point is represented by two elements (i.e., z and t) in CLR space rather than three elements in (x, y, t) space, reducing storage space for each control point by 33%.

CLR space shows a significant storage advantage for the space-time prisms as well (see Figure 15). For example, storing space-time prisms of 10,000 mobile phone users in CLR space required 1259.77 MB, or 33% of the storage space required for the same dataset in (x, y, t) space. This storage efficiency is due to the same reasons as the space-time path case. For example, 82,757,434 control points at network nodes (49.88%) and link intermediate vertices (50.12%) were needed to store the space-time prisms of 10,000 users in (x, y, t) space, whereas only the first of these are needed in CLR space.

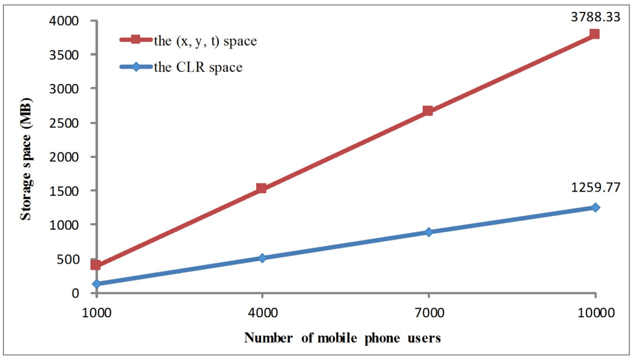


Figure 15. Storage space of space-time prisms

5.3. Spatiotemporal query performance

This section examines the performance of the spatiotemporal queries presented in Section 4.3. All reported query performances were the average of ten runs. Due to the lack of corresponding query implementation in (x, y, t) space in the literature, only queries in CLR space were examined. The query performance includes two components: the computational time for searching entities in the spatiotemporal indexes and the I/O time for retrieving the resultant spatiotemporal entities from the database. We cannot separate these two components, because we employed the ArcEngine development kit which integrates the components in the spatial query function. However, generally the first component is much smaller than the second.

Figure 16 shows the performance of the path-path intersection query: find all space-time paths of taxis intersecting the input path. To test the query performance under different numbers of taxis, five datasets were extracted from the original floating car dataset. The input path was the space-time path of a taxi randomly selected from the floating car dataset and used for all testing datasets. The performance degraded with the number of taxi cabs. For example, 0.71 seconds was required to perform a path-path intersection query in the dataset of 1,000 taxis, whereas the query grew to 8.57 seconds for the full dataset with 13,445 taxis.

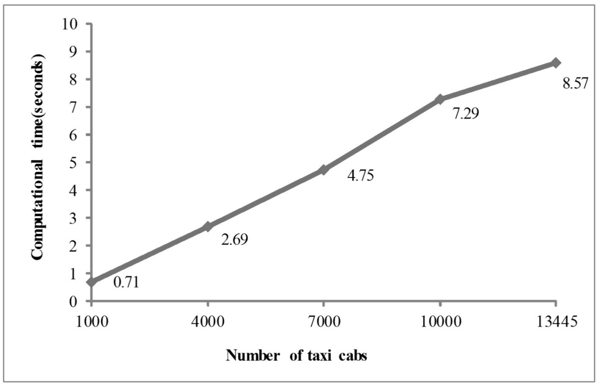


Figure 16. The path-path intersection query performance

Figure 17 illustrates the performance of the window-path intersection query using the same five floating car datasets. The window-path intersection query is to find all space-time paths existing in the input spatial window W = (x_1, x_2, y_1, y_2) at any time instance during the input time period $[t_o, t_d]$. The query performance was examined in terms of three parameters: size of the spatial window, size of the time period, and the number of taxis in the dataset. The size of the input spatial window was measured by the number of network links intersecting the window. In all experiments, the starting time of the input time period was set as 8 a.m. and the length of time period was determined by the time interval. The performance of the window-path intersection query degraded with increasing of taxi number in the dataset. However, the query performance is reasonable for this typical large-scale dataset of 13,445 taxis, requiring approximately 40 seconds when the size of spatial window was set to 10,000 links and time interval 60 minutes.

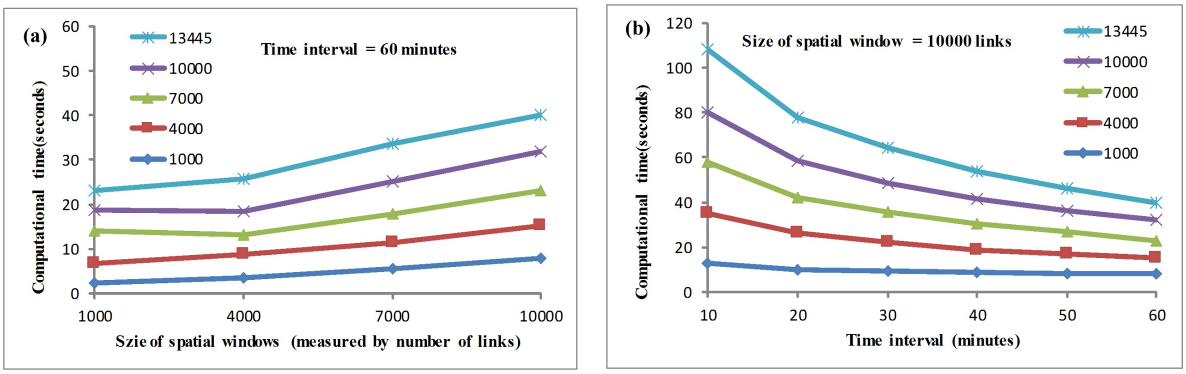


Figure 17. The window-path intersection query performance

Figure 17(a) shows the computational performance of the window-path intersection query under different window sizes. The time interval was set to 60 minutes. The size of input spatial window has a significant impact on performance. For example, the window-path intersection query required 23 seconds on the dataset of 13,445 taxis with spatial window 1,000 links, whereas this increases to approximately 40 seconds (1.73 times) when a spatial window of 10,000 links was used. This result was expected. The window-path intersection query employed a traditional spatial query on the Shenzhen road network to determine all links $a_u,...,a_v$ intersecting the input spatial window W. Then, a space-time region was constructed in CLR space by generating the set of space-time polygons $O\%_u,...,O\%_v$ for all the links $a_u,...,a_v$. As with a traditional spatial query, the larger spatial window tended to require longer spatial query times. In addition, the more number of links intersecting the input spatial window, the more space-time polygons were generated to conduct the spatiotemporal query in CLR space, thus leading to the longer query time to find the space-time paths.

Figure 17(b) shows the computational time required by the window-path intersection query under different time intervals. The size of spatial window was set as 10,000 links. It is interesting that the performance improved with increasing time interval. For example, when the time interval was set 10 minutes, the query required approximately 108 seconds to the space-time paths on the dataset of 13,445 taxis, whereas when the time interval was increased to 60 minutes, the computational time significantly reduced by 63% to 40 seconds. This result may be due to the following reason. The input spatiotemporal window comprises a set of disjoin spatiotemporal polygons in CLR space. The increase of time interval would enlarge the sizes of spatiotemporal polygons, and increase the chance of spatiotemporal intersection with the taxi space-time paths. If the spatiotemporal intersection is determined between a spatiotemporal polygon and a taxi space-time path, then the spatiotemporal intersection relation will not be checked by the taxi space-time path with other spatiotemporal polygons of the input spatiotemporal window. Therefore, the increase of time interval would reduce the computational cost in determining the spatiotemporal intersection relation.

Figure 18 shows the performance of the station-prism within query and prism-prism intersection queries on the mobile phone tracking dataset. To test the performance of these two queries under different numbers of users, five datasets were extracted from the original dataset.

The station-prism within query was to find all mobile phone users feasible to conduct an activity at a given location z_i^u during the a time period $[t_o, t_d]$. A space-time station on the input location and time period was constructed and the spatiotemporal query was implemented as a spatial within query to find users' space-time prisms containing the constructed station. The time period was set from 8–8:30 a.m. and the location z_i^u was randomly selected from the set of POIs (points of interesting) in Shenzhen city. As shown in the figure that the performance of station-prism within query were reasonable for all five datasets. For example, the query required approximately 11 seconds for the dataset of 1000 users and 108 seconds for the dataset of 10,000 users.

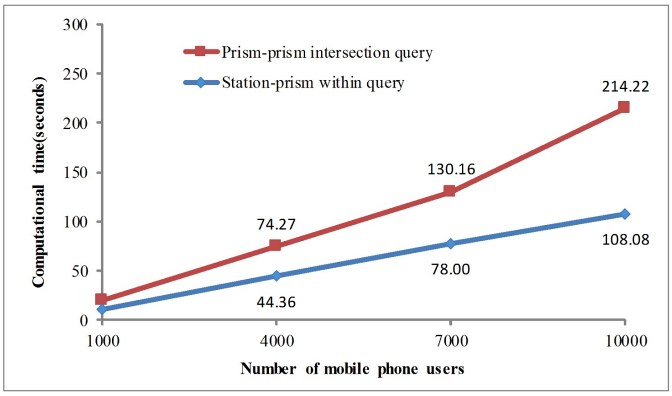


Figure 18. Computational performance of the station-prism within query and the prism-prism intersection query

The prism-prism intersection query was to find all mobile phone users feasible to conduct a joint activity with a given individual. Using the space-time prism of the given individual as the input, this spatiotemporal query was implemented as a spatial intersection query to find users' space-time prisms intersecting the input prism. The input space-time prism was the space-time prism of a mobile phone user randomly selected from the mobile phone tracking dataset. As illustrated in Figure 18, the performance of prism-prism intersection query reasonable for all five datasets. It required approximately 214 seconds for the dataset of 10,000 users. This spatiotemporal query required more computational time than the station-prism within query because the input space-time prism generally covered a large space-time region with several disjoint space-time polygons in CLR space.

6. Conclusions

This study proposed a spatiotemporal data model for time geographic entities and relations in road networks. The proposed spatiotemporal data model was built using the compressed linear reference technique. Network time geographic entities (including space-time path, station, prism and lifeline) were transformed from 3D (x, y, t) space to 2D CLR space. The equivalence of representing network time geographic entities in CLR space and classical (x, y, t) space was rigorously proved. Space-time intersection relations were refined into five space-time sub-relations (i.e. Contains, Within, Touches, Overlaps and Crosses) and implemented by the classical spatial topological relations in CLR space. Path bundling was formulated and solved as a space-time intersection problem using spatiotemporal buffering. Using the proposed spatiotemporal data model, four spatiotemporal queries were developed: path-path intersection query, window-path intersection query, station-prism within query, and prism-prism intersection query.

Compared to the classical data model in (x, y, t) space, the proposed spatiotemporal data model has several unique features.

1. Using the proposed spatiotemporal data model, the network time geographic entities can be stored and managed in classical spatial databases. This makes it easy to integrate time geographic analysis with the powerful visualization and spatial analysis capabilities provided by contemporary GIS platforms.
2. Numerous efficient spatial operations can be directly utilized to implement spatiotemporal operations for network time geographic entities in CLR space. This can lead to more efficient time geographic analysis, since it is always computationally simpler to manipulate entities in 2D space than 3D space.
3. Existing well-developed spatial index structures can be directly employed to implement spatiotemporal queries on the network time geographic entities in CLR space. This can lead to efficient spatiotemporal queries on large-scale time geographic datasets.

To validate the proposed spatiotemporal data model, a prototype system was implemented using the ArcGIS spatial database and development kit. A case study using real space-time paths of taxis and space-time prisms of mobile phone users was performed on the Shenzhen road network. The proposed model provides 54% and 67% storage space reductions for space-time paths and prisms respectively, compared with the classical data model. Spatiotemporal queries on the prototype CLR space system showed reasonable performance for large-scale time geographic datasets.

Due to space limitations, the computational experiments presented in this article are by no means comprehensive. In the case study, the Grid-File index was employed and examined for the spatiotemporal queries of space-time paths and prisms. Other well-developed spatial indexes, such as R-tree and Quad-tree, can also be used as the spatiotemporal index in the CLR space. Which spatial index techniques perform best in the CLR space needs further investigations. In the case study, the storage and query performances of space-time paths in the CLR space were examined using the ArcSDE and Oracle techniques. Comparing such performances of space-time paths with the existing moving objects databases techniques (Güting et al., 2006; de Almeida and Güting, 2005) is required in the future studies.

Several directions for future research are worth noting. Firstly, the proposed spatiotemporal data model has only been applied to time geographic entities in a single-mode road network. The extension to multi-mode networks (e.g. bus and subway) is an interesting topic for further work. The extension of the proposed model to the planar space, where people can move freely in the geographical space, also needs further studies. Secondly, ArcSDE and Oracle were adopted in the case study as the spatial database. The integration of emerging non-SQL database techniques (e.g. Mongo DB) may be a possible way to further improve the performance of spatiotemporal queries. Last but not least, this study presented four simple spatiotemporal queries for illustration, which clearly needs to be extended. The application of the proposed spatiotemporal data model for human mobility studies and advanced time geographic analysis is warranted for further study.

Acknowledgments

The authors are thankful to the anonymous referees for their comments and suggestions that improved this article.

References

Armstrong, M. P. 1988. Temporality in spatial databases. In: Proceedings of GIS/LIS'88, pp. 880-889.

Charleux, L., 2015, A modification of the time-geographic framework to support temporal flexibility in ‘fixed’ activities. *International Journal of Geographical Information Science*, 29, pp. 1125-1143.

Chen, B.Y., Lu, J.Z., Wai, O.W.H. and Chen, X.L., 2012, Development of dynamic three-dimensional coastal information system: a case study in Hong Kong. *Journal of Hydroinformatics*, 14, pp. 815-828.

Chen, B.Y., Li, Q.Q., Wang, D.G., Shaw, S.L., Lam, W.H.K., Yuan, H. and Fang, Z.X., 2013, Reliable space-time prisms under travel time uncertainty. *Annals of the Association of American Geographers*, 103, pp. 1502-1521.

Chen, B.Y., Yuan, H., Li, Q.Q., Lam, W.H.K., Shaw, S.-L. and Yan, K., 2014, Map matching algorithm for large-scale low-frequency floating car data. *International Journal of Geographical Information Science*, 28, pp. 22-38.

Chen, B.Y., Yuan, H., Li, Q.Q., Lam, W.H.K. and Shaw, S.-L. 2015. Space-time path bundling based on spatiotemporal buffering operations. *Working paper*.

Chen, J., Shaw, S.L., Yu, H.B., Lu, F., Chai, Y.W. and Jia, Q.L., 2011, Exploratory data analysis of activity diary data: a space-time GIS approach. *Journal of Transport Geography*, 19, pp. 394-404.

de Almeida, V.T. and Güting, R.H., 2005, Indexing the trajectories of moving objects in networks. *Geoinformatica*, 9, pp. 33-60.

Egenhofer, M. D. and Herring, J., 1990, A mathematical framework for the definition of topological relationships. *Proceedings of the Fourth International Symposium on Spatial Data Handling*, 803-813. Zurich, Switzerland.

Erwig, M., Güting, R. H., Schneider, M. and Vazirgiannis, M., 1999, Spatio-temporal data types: An approach to modelling and querying moving objects in databases. *GeoInformatica*, 3, pp. 265-291.

Fang, Z.X., Tu, W., Li, Q.Q. and Li, Q.P., 2011, A multi-objective approach to scheduling joint participation with variable space and time preferences and opportunities. *Journal of Transport Geography*, 19, pp. 623-634.

Fang, Z.X., Shaw, S.L., Tu, W., Li, Q.Q. and Li, Y.G., 2012, Spatiotemporal analysis of critical transportation links based on time geographic concepts: a case study of critical bridges in Wuhan, China. *Journal of Transport Geography*, 23, pp. 44-59.

Finkel, R. and Bentley, J.L., 1974, Quad Trees: A data structure for retrieval on composite keys. *Acta Informatica*, 4, pp. 1-9.

Goodchild, M.F., Yuan, M. and Cova, T.J., 2007, Towards a general theory of geographic representation in GIS. *International Journal of Geographical Information Science*, 21, pp. 239-260.

Goodchild, M.F., 2013, Prospects for a space-time GIS. *Annals of the Association of American Geographers*, 103, pp. 1072-1077.

Güting, R.H. and Schneider, M., 2005, *Moving objects databases*. San Francisco: Morgan Kaufmann.

Güting, R.H., de Almeida, V.T. and Ding, Z.M., 2006, Modeling and querying moving objects in networks. *The VLDB Journal*, 15, pp. 165-190.

Guttman, A., 1984, R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD Conference 1984*, B. Yormark (Ed.), pp. 47-57 (Boston, Massachusetts, USA.

Hägerstrand, T., 1970, What about people in regional science? *Papers in Regional Science*, 24, pp. 7-24.

Herring, J., 2011, Open GIS implementation specification for geographic information - simple feature access - part 1: Common architecture, Available online at: http://www.opengeospatial.org/standards/sfa.

Huang, B., Wu, Q. and Zhan, F.B., 2007, A shortest path algorithm with novel heuristics for dynamic transportation networks. *International Journal of Geographical Information Science*, 21, pp. 625-644.

Huang, B. and Wu, Q., 2008, Dynamic accessibility analysis in location-based service using an incremental parallel algorithm. *Environment and Planning B*, 35, pp. 831-846.

Kuijpers, B. and Othman, W., 2009, Modeling uncertainty of moving objects on road networks via space-time prisms. *International Journal of Geographical Information Science*, 23, pp. 1095-1117.

Kwan, M.P., 2000, Interactive geovisualization of activity-travel patterns using three-dimensional geographical information systems: a methodological exploration with a large data set. *Transportation Research Part C*, 8, pp. 185-203.

Kwan, M.-P. and Weber, J., 2003, Individual accessibility revisited: Implications for geographical analysis in the twenty-first Century. *Geographical Analysis*, 35, pp. 341-353.

Kwan, M.-P., 2013, Beyond space (as we knew it): Toward temporally integrated geographies of segregation, health, and accessibility. *Annals of the Association of American Geographers*, 103, pp. 1078-1086.

Kwan, M.-P. and Neutens, T., 2014, Space-time research in GIScience. *International Journal of Geographical Information Science*, 28, pp. 851-854.

Langran, G. and Chrisman, N.R., 1988, A framework for temporal geographic information.

Cartographica, 25, pp. 1-14.

Li, D., Shan, J., Shao, Z., Zhou, X. and Yao, Y., 2013, Geomatics for smart cities: Concept, key techniques, and applications. *Geo-spatial Information Science*, 16, pp. 13-24.

Li, Q.Q., Chen, B.Y., Wang, Y. and Lam, W.H.K., 2015, A hybrid link-node approach for finding shortest paths in road networks with turn restrictions. *Transactions in GIS*, DOI:10.1111/tgis.12133.

Li, X. and Lin, H., 2006, Indexing network-constrained trajectories for connectivity-based queries. *International Journal of Geographical Information Science*, 20, pp. 303-328.

Long, J.A. and Nelson, T.A., 2013, A review of quantitative methods for movement data. *International Journal of Geographical Information Science*, 27, pp. 292-318.

Miller, H.J., 1991, Modelling accessibility using space-time prism concepts within geographical information systems. *International Journal of Geographical Information Science*, 5, pp. 287-301.

Miller, H. J. and Shaw, S. L., 2001, *Geographic information systems for transportation: principles and applications*. New York: Oxford University Press.

Miller, H.J., 2005, A measurement theory for time geography. *Geographical Analysis*, 37, pp. 17-45.

Miller, H.J. and Bridwell, S.A., 2009, A field-based theory for time geography. *Annals of the Association of American Geographers*, 99, pp. 49-75.

Nakaya, T., 2013, Analytical data transformations in space-time region: Three stories of space-time cube. *Annals of the Association of American Geographers*, 103, pp. 1062-1071.

Neutens, T., Van de Weghe, N., Witlox, F. and De Maeyer, P., 2008, A three-dimensional network-based space-time prism. *Journal of Geographical Systems*, 10, pp. 89-107.

Nievergelt, J., Hinterberger, H. and Sevcik, K.C., 1984, The Grid File: An Adaptable, Symmetric Multikey File Structure. *ACM Transactions on Database Systems*, 9, pp. 38 - 71.

Pei, T., Sobolevsky, S., Ratti, C., Shaw, S.-L., Li, T. and Zhou, C., 2014, A new insight into land use classification based on aggregated mobile phone data. *International Journal of Geographical Information Science*, 28, pp. 1988-2007.

Peuquet, D.J. and Niu, D.A., 1995, An event-based spatiotemporal data model (ESTDM) for temporal analysis of geographical data. *International Journal of Geographical Information Systems*, 9, pp. 7-24.

Popa, I.S., Zeitouni, K., Oria, V. and Kharrat, A., 2015, Spatio-temporal compression of trajectories in road networks. *Geoinformatica*, 19, pp. 117-145.

Richardson, D.B., Volkow, N.D., Kwan, M.-P., Kaplan, R.M., Goodchild, M.F. and Croyle, R.T., 2013, Spatial turn in health research. *Science*, 339, pp. 1390-1392.

Shaw, S.L. and Yu, H.B., 2009, A GIS-based time-geographic approach of studying individual activities and interactions in a hybrid physical-virtual space. *Journal of Transport Geography*, 17, pp. 141-149.

Song, C.M., Qu, Z.H., Blumm, N. and Barabasi, A.L., 2010, Limits of predictability in human mobility. *Science*, 327, pp. 1018-1021.

Versichele, M., Neutens, T., Claeys Bouuaert, M. and Van de Weghe, N., 2014, Time-geographic derivation of feasible co-presence opportunities from network-constrained episodic movement data. *Transactions in GIS*, 18, pp. 687-703.

Wang, D.G. and Cheng, T., 2001, A spatio-temporal data model for activity-based transport demand modelling. *International Journal of Geographical Information Science*, 15, pp. 561-585.

Wolfson, O., Xu, B., Chamberlain, S. and Jiang, L., 1998, Moving object databases: Issues and solutions. In: *Proceedings of the 10th International Conference on Scientific and Statistical Database Management*, pp. 111-122.

Xu, J.Q. and Guting, R.H., 2013, A generic data model for moving objects. *Geoinformatica*, 17, pp. 125-172.

Yu, H. and Shaw, S.L., 2008, Exploring potential human activities in physical and virtual spaces: a spatio-temporal GIS approach. *International Journal of Geographical Information Science*, 22, pp. 409-430.

Yuan, M., 1999, Use of a three-domain representation to enhance GIS support for complex spatiotemporal queries. *Transactions in GIS*, 3, pp. 137-159.

Yue, Y., Lan, T., Yeh, A.G.O. and Li, Q.-Q., 2014, Zooming into individuals to understand the collective: A review of trajectory-based travel behaviour studies. *Travel Behaviour and Society*, 1, pp. 69-78.