GenerativeMap: Visualization and Exploration of Dynamic Density Maps via Generative Learning Model

Chen Chen, Changbo Wang, Xue Bai, Peiying Zhang, and Chenhui Li

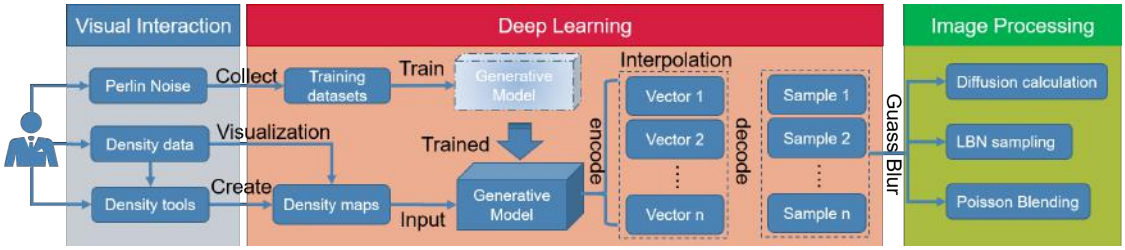


Fig. 1. A definition of GenerativeMap. We design a general pipeline for visualization and exploration of the dynamic density map. The whole pipeline is composed of three modules in the definition, and the complete process is described in Section 3.

Abstract—The density map is widely used for data sampling, time-varying detection, ensemble representation, etc. The visualization of dynamic evolution is a challenging task when exploring spatiotemporal data. Many approaches have been provided to explore the variation of data patterns over time, which commonly need multiple parameters and preprocessing works. Image generation is a well-known topic in deep learning, and a variety of generating models have been promoted in recent years. In this paper, we introduce a general pipeline called GenerativeMap to extract dynamics of density maps by generating interpolation information. First, a trained generative model comprises an important part of our approach, which can generate nonlinear and natural results by implementing a few parameters. Second, a visual presentation is proposed to show the density change, which is combined with the level of detail and blue noise sampling for a better visual effect. Third, for dynamic visualization of large-scale density maps, we extend this approach to show the evolution in regions of interest, which costs less to overcome the drawback of the learning-based generative model. We demonstrate our method on different types of cases, and we evaluate and compare the approach from multiple aspects. The results help identify the effectiveness of our approach and confirm its applicability in different scenarios.

Index Terms—Density map, deep learning, spatiotemporal data, generative model

1 Introduction

The spatiotemporal datasets collected by sensors have become larger and cover more research fields in recent years. Exploring the dynamic of data is a long-term challenge; the collected spatiotemporal data are often discrete and static, and we can obtain one series of states in the scenario through visualization. In traditional work, users can analyze the distribution and time-varying patterns based on data features. However, to find more details by observing these data, the dynamic process is important, and a natural representation of data movements helps users to understand the relationship between states. Furthermore, many existing spatiotemporal data lack certain parts of records, and the interpolation approach can help to fill the record gaps.

The density map is a simple presentation and available for many types of data. The application examples contain data mapping in meteorology, movement of flow in oceanography, geographical distribution of people location, etc. Currently, researchers apply this technology to detect and track patterns from the ensemble dataset, which is a collection of spatio-temporal results [42]. The density map is also valuable in the simulation field and is often combined with fluid simulation or data sampling. However, the density map is vague, and it is difficult to analyze the features of the image by classical image processing methods. The density map often consists of a large number of data points, and the final visual effect shows the data cluster, which means that the generated rules are often hidden in the formation process. For these reasons, it is meaningful and necessary to explore dynamics on the density maps.

Figure 2 describes the issue that we want to address. First, density maps can be manually created by some rules. The process of artificial creation is given in Figure 2(a), which serves as the simulation and visualization. Can we guess the probable change process if we only know a few rules? Second, some researchers generate a large number of visualizations with scientific data, which often occurs with many types of research [38, 48]. Figure 2(b) shows that researchers want to know the continuous process, although there is less context information. How can we describe the possible change in the data? Third, as shown in Figure 2(c), large images are used to contain large-scale information, while users only focus on certain parts (we call these boxed images) [35]. What happens in these parts that we focus on when other parts contain less information? In different domains, there are many works to solve these problems. However, the complex model and costly computer implementations are limitations of related approaches, and a simple general method can help solve these problems.

Compared with previous methods, we take advantage of generative models. Generative models provide many novel solutions for data generation in recent years and show excellent performance regarding nonlinear and self-learning. As two major categories of generative models, the variational autoencoder (VAE) [22] and generative adversarial networks (GANs) [16] have both shown advantages. The encoder provides a mapping relationship between the image and code, which enables users to establish the feature space. GANs are a popular model for generating high-resolution and photorealistic images, and there are many improved models for different tasks. In our work, we introduce a GAN framework to implement the smooth morphing of density maps, and the approach needs fewer parameters without any human involvement. A known drawback of GANs is that the generating performance

• *C. Chen, C. Wang, X. Bai, P. Zhang, and C. Li are with the School of Computer Science and Technology, East China Normal University. Chenhui Li is the corresponding author. E-mail: chli@sei.ecnu.edu.cn.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

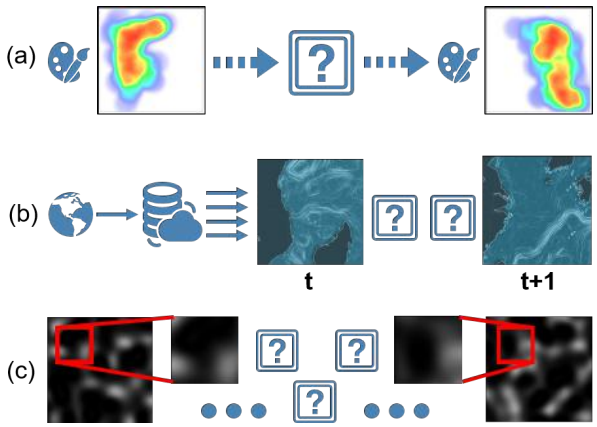


Fig. 2. Explanation of problems scenarios that may arise. The three parts are typical applications of density map evolution, they are the main motivation of our work. The details are introduced in Section 1.

is low for large images, and it is usually computationally complex and time consuming. As an improvement, for the region of interest (ROI), we apply image fusion to solve the problem. Another important work is the information visualization of movements, which helps users to correctly identify the change or trend of the dynamic data. A typical method is vector field representation, and the results are often shown by points, lines, and textures. We improve the visual quality and present the direction of a field by adopting blue noise sampling. The technology overcomes the visually disturbing aliasing artifacts and is better defined to our perception. This approach only needs a few parameters to adjust to different applications. For providing a more usable visual experience, we improve the algorithm with the level of detail (LOD). The final result shows a smooth and user-friendly representation.

Our approach can be combined with many methods of image processing and graphics, providing a novel pipeline that we call GenerativeMap. The method aims to visualize the dynamic change of two density maps in a convenient form. GenerativeMap is not meant to replace the dynamic analysis method in other domain fields; rather, it is more of a complementary method for these works, particularly for cases where the data are missing or the state is too uncertain to estimate. Crucially, this method is an attempt to improve visualization with deep learning, and the pipeline can be extended by continuous development of generative models. To achieve a generic technique, we simplified the operation by setting as few user parameters as possible. With traditional approaches, users need to change many parameters and the framework to deal with different tasks. In our work, we introduce several technologies to overcome these disadvantages, and users just need to replace the datasets if the trained model is not suitable. As shown in the experimental section, we apply the proposed approaches to the artificial datasets, eddy datasets, and location-based datasets, which are all based on the density map. The cases verify that GenerativeMap is a general pipeline that can extract the dynamics of the density map effectively, and the results are smooth and continuous. In our work, the visual effect and generality are the important points of the work. Our main contributions are summarized as follows:

A novel pipeline for exploring the dynamic of a density map: A general pipeline can be applied to design several tools, combined with advantages of deep learning and visualization, such that the results help users to identify dynamic evolution in different scenarios effectively.

A generative model for extracting the dynamics from two discrete images: An improved generative model is promoted to compute the probable dynamic change of images, which is available for many types of datasets. To employ GANs in the large image, we further combine Poisson blending to improve the visual effect, which avoids time-consuming data loading.

A sampling method for presenting distribution change: To identify the distribution of the density and the dynamic trend of evolution, on the basis of blue noise, the proposed sampling method is combined with LOD, which can enhance visual perception particularly for ROI.

2 Related Work

Visualization of a dynamic density map is an area in which extracting data features have mostly been implemented by visual analysis. Users can employ their methods to compute the evolution process or forecast potential states, and domain experts help developers to analyze tasks and cases. These kinds of extracted work are often built based on physical or mathematical models, which focus on the accuracy and explanation of the results. To obtain the interpolation of two density maps, the work presented in this paper is related to three broad topics: 1) spatiotemporal dynamic extraction, 2) realistic image generation, and 3) movement field visualization.

2.1 Spatiotemporal Data Extraction

In recent years, a large number of spatiotemporal data are produced and collected, particularly in fields such as meteorology, oceanography, and transportation. As classical tasks, the extraction, dynamics and fusion of ensemble datasets are all topics that have been researched [33]. The topology methods can extract the time-dependent vector fields [18]. These types of methods can also be extended by other scene-related approaches that can be widely used to solve problems such as detection and tracking [39]. Similar methods have contributed solutions for the movement analysis of ensemble dataset, which has been verified by many experiments [13,28]. These cases show that applications currently exist that can partly extract the change process of spatiotemporal data.

In addition to the feature-based methods above, Ayan et al [2] proposed a visualization method to analyze the weather ensembles. Distance-based approaches and the projection transform were also used to meet this requirement in recent years, and the similarity measure and the dimensionality reduction are the core steps in these works [17,51]. In the traditional gap-filling field, researchers attempt to extract the nonlinear spatio-temporal patterns with a data-driven method such as neural network [34]. Nowadays, the deep learning approach can make a general model available for more scenarios [23] according to the intersection of different disciplines.

2.2 Generative Learning Model

Deep learning has made great advances in developing generative models, which can transform data between a simple latent distribution and a complex distribution. Three common generative learning models are GANs, VAE and the flow-based model. GANs are used to generate realistic high-quality images by discriminating the random fake distribution from the real distribution of training datasets [16]. However, the latent distribution is assumed as random in GANs; this characteristic makes GANs unstable, and the high-resolution image generation is a challenge for GANs. VAE is a framework that encodes images as a definitive latent vector space [22]. The outstanding feature of the method is that it takes the target image as a posterior distribution, although the classical VAE can only obtain approximate results. The flow-based model is a mathematically based method, and it has solved image generation by data space mapping [11]. The related idea can generate natural and high-definition results; however, its computational cost is high.

In fact, deep learning is usually used for interpolating discrete density maps. The results proved that linear interpolation in latent space will lead to linear change generation of features [32,45]. Other important research focused on how the interpolation methods influence the continuous change. Piotr et al. [3] tried many experiments and concluded that the linear interpolation could present enough features when the data are low-dimensional. Taking advantage of VAE and GAN, the hybrid model can encode images as definite vectors distributions and generate high-resolution images. Typical hybrid models are VAEGAN, ALI and BiGAN [12, 14, 25]. Our work is based on encoding ability and interpolation validity of these type deep learning models.

2.3 Movement Field Visualization

There has been much visualization research to present density movement. In previous work on scientific visualization, points, lines, and textures often represent vector fields [4]. The information can be clearly

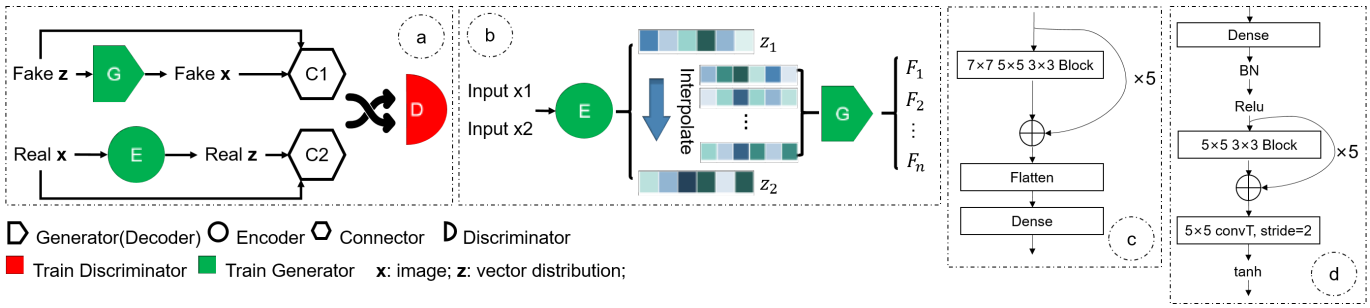


Fig. 3. The structure of our improved BiGAN model.(a) The framework of network; (b) The interpolation of images; (c) The structure of encoder; (d) The structure of decoder.

indicated in a suitable glyph, and then, the results can indicate the evolution process or the relationship among datasets [10, 41]. For analyzing spatiotemporal data, describing the varying process of events is a common task. Inspired by flow visualization, novel visual methods help users to identify the flow direction of information based on relating models [21, 46]. All these experiments determined the meaning of extracting the dynamic change of data. The image method has attracted more attention in recent years, and the spatial dimensions can be transformed to show states and movements [6]. This is an important ability currently because data collections are often discrete, and a probable reference can help users infer more information without data.

The ensemble movement description and forecast are promoted with the big data collection [27, 29] to improve the work above. Uncertain visualization in these domains confirms that a probable data inference is meaningful for experts, which is also an important motivation of our research. Blue noise is widely applied in dynamic sampling and stippling patterns since uneven distributions are encountered, and it can avoid aliasing artifacts [47]. The technology can help users identify multiple classes and spatial positions of samples by extending the original methods [9, 43], the results have worked well for improving visual effects and domain problems. The related visualization work also confirms that the glyph, e.g. arrow, can present the change trend [30].

3 GenerativeMap Pipeline

Most visualization methods of dynamic data combine feature analysis and physical modeling, and these works often contain requirements analysis and clear tasks from domain experts. Such ideas have proven to be available for special tasks in many works. However, such extracted data variation needs a large number of prework steps and has a complex data preprocess, and users commonly need to set many parameters. In addition, most evolution estimation methods are designed for special tasks, it is difficult to apply an existing method into other scenarios. Furthermore, the traditional pixel-based image processing approach has a few limitations, particularly for the target that moves a great distance or crosses with multiple source kernels.

In this paper, we propose GenerativeMap to show the morphing between two density maps, which extract dynamic features by the generative model. Image interpolation is usually used to reason whether model learned relevances and representations, rather than remembering sharps as discussed in the works [1, 32, 45]. The related theories and experiments are the basis of our work. The results show that the learned space has smooth transitions. Walking in the latent space will result in the semantic changes. Compared with the most of deep learning works, we need a controllable image interpolation approach, since many GANs models generate samples randomly. As we introduced in Section 2.2, the encoder and decoder can help model to construct the relationship between training sample and latent vector. An ideal model should get two selected density map as inputs, and generate a series of smooth and continuous transition samples as outputs.

We transform visualization images into inputs of the generative model. Since parts of the initial visualization scenarios are not density maps, we design a density map tool. Users can use the tool to transform the real-world visualization scenarios into density maps. Training deep learning model is another important pre-work. We seriously consider how to create suitable training dataset for learning general dynamic features. To make the model generate smooth and continuous samples, we propose a special dataset creation method in Section 4.1.

Figure 1 shows the overview of the pipeline, and we consider GenerativeMap as a general method that can be used in multiple scenarios, as referred to in Section 1. In the first part of GenerativeMap, users choose two discrete frames at two noncontinuous times as the inputs of the module. These visualization results are presented as density maps by the density map tool. Users can upload special existing density maps as inputs. For a special domain application, the model can be trained better with special training datasets, which is not the focus of our work.

The second part is a simple and effective generative model, which is the core part of the pipeline. Smooth image transition relies on continuous latent vectors interpolation. The trained encoder could encode two input images into two definite distribution in vector form. Linear interpolations of the vector distributions will be decoded as linear output samples. We further improve the performance of the model by designing modules of the network. The model can process larger images and show more details. The modification of modules can be seen in Section 4.2.

It is not always natural when playing out interpolated samples directly, additional image processing methods are necessary for some complex conditions. High-resolution images, less information, and uneven small change, these are all weaknesses of deep learning. The high-resolution image generation is a challenge for deep learning models, so we propose to use an image fusion approach to solve the problem. Since the interpolations are discrete and the regions are small parts of the whole image, the transition between interpolations may not be remarkable in some conditions. To show the change clearly, we calculate the color gradient of images by diffusion model. Inspired by variation field visualization, the direction glyph can be used to represent the gradient change. For showing clear uneven changes, transition presentation is optimized by an improved sampling method based on a blue noise approach.

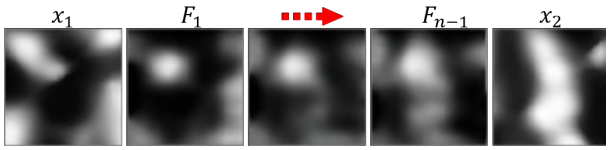
To simplify the procedure and explore a generic method, our work combines deep learning and image processing with visualization, providing an easy-to-use and comprehensive approach. We describe the details in Section 4.2 and 4.3.

4 Method Description

To train a general deep learning model to explore dynamic patterns of density maps, it is important to create available datasets. The property of samples in training data should contain different sizes, shapes, and time intervals, and they should also provide smooth and continuous dynamic information. As a visualization system, we want to balance the conciseness and computational capabilities and to introduce as few interactions as possible. The dynamic change may be difficult to be identified, particularly when there are slight changes and noises. We enhance the visual effect to improve our system by using blending, field representation, and sampling. The next parts introduce how to apply these technologies to implement a general framework. For conveniently explaining our methods, we define some symbols as listed in Table 1.

Table 1. Symbol definitions

Symbol	Description
$x1, x2$	$x1, x2$ represent the two density maps are the input of our deep learning model.
$z1, z2$	$z1, z2$ are vectors distributions of $x1, x2$, which are encoded by encoder of our model.
F_n	F_n illustrates the interpolations of our output, n means the nth samples of the continuous interpolations.
G_n	G_n describes the ground truth images, n means the nth image in the series of ground truth.



(a) Interpolation of original BiGAN

(b) Interpolation of Mp-GAN

Fig. 4. We attempt employing the same data with the original BiGAN and our Mp-GAN, and it easily shows that the result of the latter would be better.

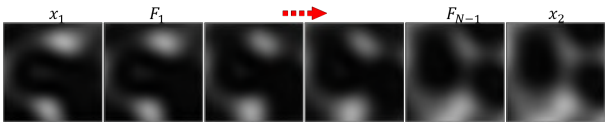
4.1 Dataset Generation

The dataset generation is an important step to obtain a useful deep learning model, since the model will learn our expected features from data. There are no existing reliable datasets for density map morphing. Initially, considering that the density map is based on the kernel density estimate (KDE) [5], we created datasets by filling images with pixels, where the pixels follow Gaussian distributions. According to the obtained results, the model can learn the Gaussian shape well and generates a series of similar figures, which demonstrates the feasibility of this idea. However, the model cannot learn the dynamic change since the samples in training datasets are independent. The ideal model should learn natural and random distributions that contain smooth and continuous features.

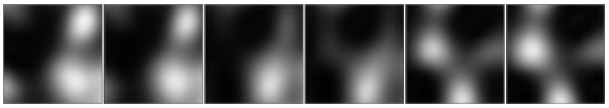
Perlin noise (PN) is an effective algorithm that is widely used in creating and simulating natural scenes in the virtual world [31]. It starts with random pixels in a 2D space, and the noise generates pseudorandom gradient vectors for every pixel. The quality of generating samples is controlled by the randomly generated seeds, the Gaussian fuzzy sets and smooth interpolation functions. The formulations of PN are as follows:

$$\begin{cases} T(t) = \alpha t^5 - \beta t^4 + \gamma t^3 \\ S_n^v = T(g_0^v - g_n^v) \\ I(x,y,w) = x(1-w) + y \cdot w \\ \Delta(g_n) = (g_0 - g_n) \cdot P[g_n](g_0 = (x,y), n = 1,2,3,4) \\ \Theta(g_0) = I(I(\Delta(g_1), \Delta(g_2), S_1^x), I(\Delta(g_3), \Delta(g_4), S_1^x), S_1^y) \end{cases} \quad (1)$$

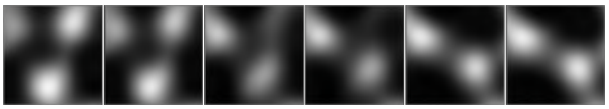
where $T(t)$ is a function that generates nonlinear movement, and α, β, and γ are all hyper parameters. t indicates the arbitrary value. g_0 means the current position of the pixel. g_1,g_2,g_3, and g_4 are the vertexes of the seperated space grid. S_n^v means the smooth weight between g_0 and g_n along v direction. $I(x,y,w)$ represents the interpolation method between x and y, which is influenced by w. $\Delta(g_n)$ is the composed gradient of g_n and $P[g_n]$ means a gradient vector random selected from a finite number of precomputed gradient vectors. $\Theta(g_0)$ is the actual movement of the pixel g_0. Gaussian fuzzy can reduce the additional visual noise.



(a) Connection



(b) Shift



(c) Fusion

Fig. 5. The generated series shows the multiple types of results with our model. According to the change process of start-end frames, we simply classify the generative form as three types.

We can get the static PN images by steps above, which contain smooth and natural shapes. However, the ideal model should generate a series of continuous and smooth interpolations to describe the transition process of two discrete density maps, which means that the deep learning model can learn the relationship between two density maps from the training datasets. Inspired by the related work [50], we use the neural network to extract expected continuous features from keyframes. The keyframes are from the dataset consist of spatiotemporal records. The data collection approach in our work can be summarized as three steps:

1. Setting parameters to generate dynamic PN images. The parameters are related to the shape, size, lightness, and scale of noise.

2. We randomly select multiple regions in the PN images and grab a fixed size of image frames at a regular time interval.

3. After getting a certain amount of data, we repeat $Step1$ and $Step2$ until we have enough data collected.

4.2 Density Generative Model

As described before, there are different ways to generate images' interpolation with a deep learning model. In this paper, we follow a BiGAN framework as the basis of our method, which contains the encoder, generator (decoder) and discriminator. The structure can encode the selected image into a specific distribution, which is the necessary condition to interpolate two target images. The core framework of BiGAN is shown in Figure 3(a), indicating that the encoder and generator (decoder) will be trained separately. The encoder transforms the real image (real x) into the real distribution (real z), while the decoder generates the fake image (fake x) by prior distributions (fake z). Then we concatenate distributions and corresponding images as pairs, the discriminator will try to distinguish if a pair is from encoder or decoder (C1 or C2). The above training method ensures the trained model can construct the relationship between input images ($x1$,$x2$) and latent vector space ($z1$,$z2$), since the network learns the image and distribution at the same time.

DCGAN, a sound and mature model, provides convolution structure reference for the unsupervised learning and confirms GAN can be used to extract features [32]. However, the original model processes 32×32 images, which is too small to be used in visualization. A classic case of results is shown in Figure 4(a), where there are shortcomings in the details, particularly for the shape of the edge.

In our work, the PN images are large, and we hope the model can learn a wider region of features. The improved components are shown in Figures 3(c,d) present the structure of the encoder and decoder, respectively, which makes the improved network applicable to the

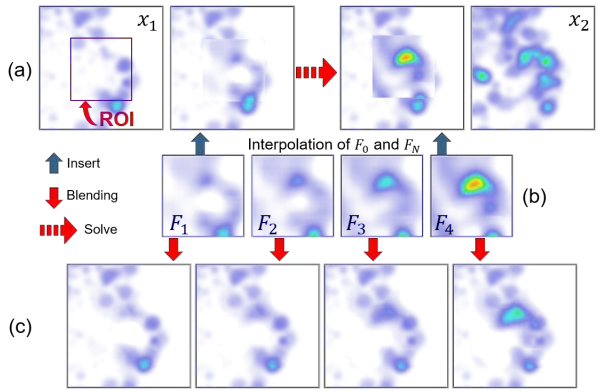


Fig. 6. The series explains why and how we should employ Poisson blending in parts of a large image.

large image, and we refer to other successful image networks such as ResNet [36]. Inspired by ResNet, we design similar blocks and select larger convolution kernels to enable the model to process a larger image with more information. In our work, we summarize that the encoder needs to collect information, so the encoder should contain more big size kernels and convolution layers (Figure 3c) and decoder employs several deconvolution layers as symmetric parts (Figure 3d).

In addition to the structure improvement, we refer to the loss function in many classical neural networks. For BiGAN training of the encoder and generator (decoder), two loss functions are designed in the approach. The final losses are described as Equation 2:

$$\begin{cases} Dloss & = max(\mu(\Delta T - \frac{\Delta T^2}{2\lambda d(x,z)})) \\ Gloss & = min(\mu[\Delta T + \beta_1|z - E1(G(z))| + \beta_2|x - G(E1(x))|]) \\ \Delta T & = T(E2(x), E1(x)) - T(E2(G(z)), z) \\ d(x,z) & = \mu((x - G(z)) + (E1(x) - z)) \end{cases} \tag{2}$$

where *Dloss* and *Gloss* mean the loss function of the encoder and decoder, respectively, and μ means average operation of the loss. ΔT means the difference between images and distributions, which is calculated by discriminator. $d(x,z)$ is an operator measure distance between two sets, and differences between images x and distributions z are calculated in this paper. $E1$, $E2$ and G represent the encoder for generating, the encoder for discriminating and the generator, respectively. x, z are symbols of real images and fake distributions, respectively, as shown in Figure 3(a). λ, β_1 and β_2 are all hyperparameters to be set in our experiment. The interpolation results are shown in Figure 4(b), the shape of which is similar to the inputs, and the morphing result is smoother than the result in Figure 4(a).

As the key step, we use the generator of the model to get the interpolation capability, and the Figure 3(b) present the detail. Users select two density map as the real images x_1 and x_2. The trained encoder will encode these images into two vector distributions z_1 and z_2. After z_1 and z_2 are interpolated, the trainable decoder can decode these interpolations into images set $\{s_1, s_2 ... s_n\}$. According to the related deep learning work, the set contains continuous features and the linear interpolation will present the smooth transition [45]. We further evaluate the generative ability and the interpolation quality in Section 6.

After the above steps, we ultimately construct a generative model for a 128×128 image, and most models can achieve a good performance with this size. To distinguish it from the original BiGAN framework, we call it Morphing GAN (Mp-GAN). Our model is designed for morphing between density maps. We classify the dynamic process as three types: connection, shift, and fusion abilities. Figure 5 shows the generated results, in which the interpolations are smooth and clear. In our work, both a complex network model and an excessive computational overhead are unavailable characteristics for creating an easy-to-use method. One important aspect to circumvent these issues is that users can focus on parts of the image rather than the full information in many scenarios; the other factor is that images may contain some valueless

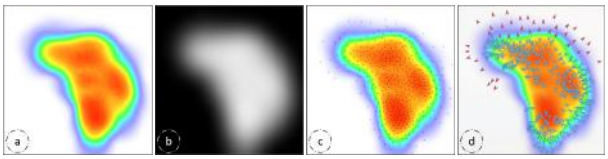


Fig. 7. The experiment shows the visual effect of blue noise sampling. (a) Colorful density map; (b) Gray density map; (c) Blue noise sampling result; (d) Field variation representation.

information. Considering these reasons, it is not worthwhile to design a complex special model for the large-sized image task.

Figure 6 demonstrates a typical scenario, and a large density map illustrates a large range of the information distribution. The selected rectangle is the ROI where information is collected, and it is obvious that the uneven distributions in this region would change in the density map. Initially, we try to obtain the part of an image by image segmentation and then insert the generated interpolation into the source image directly, as shown in Figure 6(a). The density change is unnatural, particularly on the boundary.

Poisson blending is an approach to blend parts of an image into another one and smooths the boundary of images. The challenge has been solved by taking images as functions, and blending means minimizing the difference between two functions. The core equation can be described as Equation 3:

$$\begin{cases} \Gamma_N H(x,y) = \phi(H) + \psi(A) + \chi(B) \\ \phi(H) = \sum\limits_{(dx,dy)+(x,y)\in\Omega} H(x+dx, y+dy) \\ \psi(A) = \sum\limits_{(dx,dy)+(x,y)\in\partial\Omega} A(x+dx, y+dy) \\ \chi(B) = \sum\limits_{(dx,dy)+(x,y)\in\Omega\cup\partial\Omega} (B(x+dx, y+dy) - B(x,y)) \end{cases} \tag{3}$$

where $\Gamma_N H(x,y)$ means N points around digital (x,y) in the new merged image H, A is the target image and B is the source image. (dx,dy) is the probable position of the adjacent pixel, Ω represents the area of B, while $\partial\Omega$ is the boundary of A. To summarize, we define $\phi(H)$, $\psi(A)$, and $\chi(B)$ to demonstrate the area in H, the boundary in A and both elements in B, respectively.

The interpolations are shown in Figure 6(b), and the process is smooth and continuous as in other cases. The final visual effect of processing of the image with Poisson blending is shown in Figure 6(c), and the result smooths the evolution of the density map and compensates for the disadvantage of deep learning. We can easily find that the density gradually increases in this region, where it is not influenced by density in the surroundings.

4.3 Field Variation Representation

After the target images are processed by the generative model, we obtain a series of interpolations of the selected image. However, there are two limitations if we provide these interpolated images to users directly. On the one hand, the model generates images based on the encoding vectors of targets, which are asymmetrical sometimes; therefore, the transition is unnatural. On the other hand, it is difficult to confirm the sampling number, for the number of samples we set would influence the continuous change. A direction visualization that helps users to summarize the trend is an important work for the reasons above.

Initially, we calculate the change in images based on the diffusion model and then draw arrows directly based on the gradient change similar to most of the related work [15]. As a part of our pipeline, the classical optical flow model can be described as Equation 4:

$$\begin{cases} min\{\int_\Omega (D_0(x) - D_1 \circ \mathbf{u})^2 dx + \lambda \int_\Omega |\nabla \mathbf{u}|^2 dx\} \\ |\nabla \mathbf{u}|^2 = |\nabla u|^2 + |\nabla v|^2 \end{cases} \tag{4}$$

where x means the pixels in the image, and $\circ$ represents a composite function. Ω is the range of whole image, D_0 and D_1 indicate two

images, which often record two states in continuous time. $\nabla \mathbf{u}$ is a function to calculate the gradient of a pixel, which is composed of the gradient along the u and v directions.

As shown in Figure 7(a), this is a sample with a dynamic noise distribution. Initially we want to use arrows to present the change trend by the diffusion model, for arrows glyph is a common and practical choice in a 2D visualization [30]. We design a display rule in this work; the red arrows in the experiment mean that the region is expanded, and the blue arrows represent the reduction in the area. However, the presented glyphs may overlap each other or be too small, which are the typical problems encountered using uniform sampling. Users need to set the sampling interval and threshold to avoid these display shortcomings.

Blue noise is widely used in texture synthesis, data sampling, and realistic rendering, and related applications contain stippling, visualization and reconstruction [26]. We employ the technology in this work for overcoming aliasing and increasing the space between clusters. Users can easily identify the shape of the density, and the sample seed distribution is less dense, which makes the arrows nonoverlapping without parameter setting. However, it is noteworthy that the brightness of gray images represents the data density, which is not always the same in the density map, as shown in Figure 7(b). The brightness change is important since the density has meaning, and it is obvious that users cannot obtain this brightness through observing the denseness of distributions. For the original blue noise, the key is the minimum acceptable distance between stochastic sampling points, which the main parameters control for the sampling distribution in the whole image.

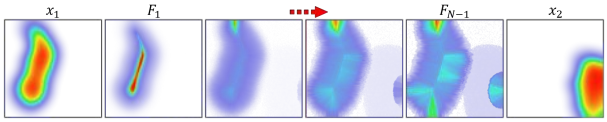
We show the uneven brightness by further enlarging the difference of distributions, which promotes the visual effect. In the simulation domain, LOD helps users to form large-scale visualization with fewer data and to quantify the sampling error [19,44]. Inspired by the solution, we classify different sample intervals as multiple levels, and the final result is merged by overlapping multiple sample distributions. The rule of the LBN distribution on the density map is built based on blue noise (BN) sampling [26]; therefore, the generated points have blue noise properties using the method. BN returns an array containing the position of samples, and the complete LBN algorithm is described as Algorithm 1. We improve the visual effect by employing a level-blue-noise (LBN) method, which combines the advantages of LOD and blue noise sampling. Figure 7(c) illustrates the sampling result using LBN, and the point indicate the sampling position. Figure 7(d) shows a case of change trend representation. Arrow size indicates the change value. Arrow direction indicates the change trend, and the arrow glyphs are distributed in the major regions.

Algorithm 1 LBN algorithm.

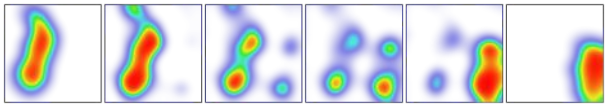
Input: P_1, P_2: the target density maps; K: the separate levels; M: the maximum value of the radius; $f_{x,y}(P)$: the significance value of digital locate (x,y) in the density map P; τ: the acceptable minimum significance value;

Output: S: The array statistic of the position of samples in all the k levels;

$(W,H) = size(P_1)$
while $k < K$ **do**
  $ck = M - f_{x,y}(P)/k$
  for $range(0,ck)$ and $(x,y) \subseteq P_1$ **do**
    $AS = [(x_0,y_0),(x_1,y_1)...(x_n,y_n)] = BN(x,y,ck)$
  end for
end while
while $(x,y) \subseteq AS$ **do**
  $x_{min} = min\{x-3*ck,0\}, y_{min} = min\{y-3*ck,0\}$
  $x_{max} = max\{x+4*ck,W\}, y_{max} = max\{y-3*ck,H\}$
  for $x_{min} \le x \le x_{max}$ and $y_{min} \le y \le y_{max}$ **do**
    $BNS = BN(x,y,ck)$
  end for
end while
while $k < K$ and $(x,y) \subseteq BNS$ **do**
  if $f_{x,y}(P) > \tau$ **then**
    $S_k \Leftarrow (x,y)$
  end if
end while
return $S = \sum_0^K S_k$;



(a) The morphing result with diffusion model



(b) The morphing result with GenerativeMap

Fig. 8. Morphing visualization of artificial data; the start frame and the end frame are created by hand-drawn methods. (a) The series are generated by the diffusion model and (b) show the smooth change by our model.

5 Experiments

We show the GenerativeMap visualization cases and the effect of the trend presentation in this section. GenerativeMap extracts the continuous interpolation of the density map and presents the morphing trends in a natural visual style. We use various common datasets and show different change trend maps. The first datasets are artificial datasets, whereas the other two are real-world datasets. All datasets are pre-rendered by Gaussian blur such that we can introduce the data into our generative model directly. Since we are focused on the transform dynamic of density maps rather than the event data following physical rules, we do not emphasize the realistic characteristics of the process details.

5.1 Artificial Data

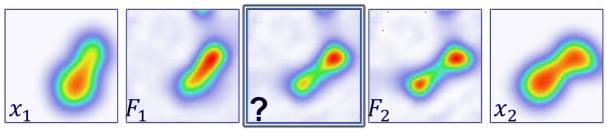
In this work, we use PN in training data for the generative model. The following experiment tests whether the model can learn easier rules in addition to noise activity, which is an important potentiality of this approach. We design hand drawing tools with a heat map and Gaussian fuzzy sets, which can provide random density as the inputs of GenerativeMap. The change process is computed by the diffusion model and GenerativeMap. This experiment illustrates the ability of extracting the change process of two unknown density maps, and we envision it as the basis of extended applications.

Figure 8(b) shows the result of an artificial data case. The input are two images, which are drawn by a volunteer. Given that they are artificial creations, in fact, no one knows the real change process. As discussed previously, in our model, the inputs are preprocessed by Gaussian blur. The middle 4 frames in the Figure 8(a) are computed by the diffusion model as a basic reference. We can find that the motion process is a challenge for the diffusion model, for there is no overlapping pixel of the input images. As a comparison, Figure 8(b) presents the interpolation results by GenerativeMap. The middle frames change smoothly and continuously, particularly the last half of the series. Based on this characteristic, the tools can be extended to other applications, such as the experiment in Section 5.2. We can also conclude GenerativeMap can help users to guess the density map change, although the learned datasets do not contain random shapes and possible change to the process.

5.2 Dynamic Eddy Data

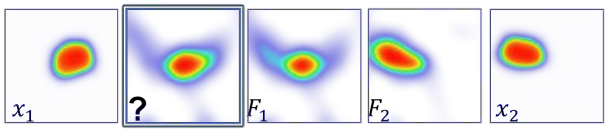
Based on the tools designed in Section 5.1, we can extend GenerativeMap in real-world scenarios. A classical density map application is eddy visualization, where the identification and tracking are the important tasks to illustrate ocean ensemble movement [7]. This work collects the eddy data in a stream type, which comes from AVISO. The site provides common flow field data combined with ocean topography

(a) The real data of two vortexes



(b) The morphing process of (a)

(c) The real data of vortex movement



(d) The morphing process of (c)

Fig. 9. We create density maps based on the eddy data, and our pipeline extracts and shows the movement of the eddy. The 1st case represents two vortexes moving toward different directions, and the 2nd case means a vortex move, where the change process shows a smooth and continuous movement.

and the satellite information, and then, we employ the LIC method to obtain a series of eddy images as our real data [8].

Using the density tools we designed above, the eddy map can easily be drawn as a density map. The original eddy maps are collected from a part of a located region in the ocean, which are shown in Figure 9(a). We mark the target regions in red boxes, and every ensemble is composed of two eddies. In traditional work, it is difficult to guess the probable move trajectory without enough data because users must analyze the movement features or summarize the basic movement rules in the ocean. However, we can obtain a basic reference by GenerativeMap without any prior knowledge. The processed inputs are presented in Figure 9(b), and as in other experiments in our work, our tools employ Gaussian fuzzy sets and gray processing in this step. A surprise to us is that the change process is not completely linear, which does not influence the results of GenerativeMap. We can find that the two eddies move toward different directions, which means that the movements of eddies are extracted independently. The pipeline still shows available results without additional supports, although the model certainly did not learn the rules to generate the vortex before.

In order to test the generative ability of GenerativeMap, we further choose a different movement type. Compared with Figure 9(a), the vortex in Figure 9(c) move from right to left. The middle three interpolations in Figure 9(d) present a natural and smooth transition, and the shadow around core can even simulate the fluid type. We also find that the generated shape of the ensemble can match the existed real data shape in the third and fourth frame.

We compare the generated interpolations with the real data frame-by-frame, and the original motion process cannot be recorded continuously because of the data property. There are only two records between the start record and the end record as shown in Figure 9(a) and (c). To obtain a continuous process, we need to make up the data records, and the generated results play well for the task. The 3rd frame in Figure 9(b) and the 2nd frame in Figure 9(d) are both conjectures, the shadow of which is the reference for the motion process. The result indicated that our model can be used for data interpolation in addition to image interpolation.

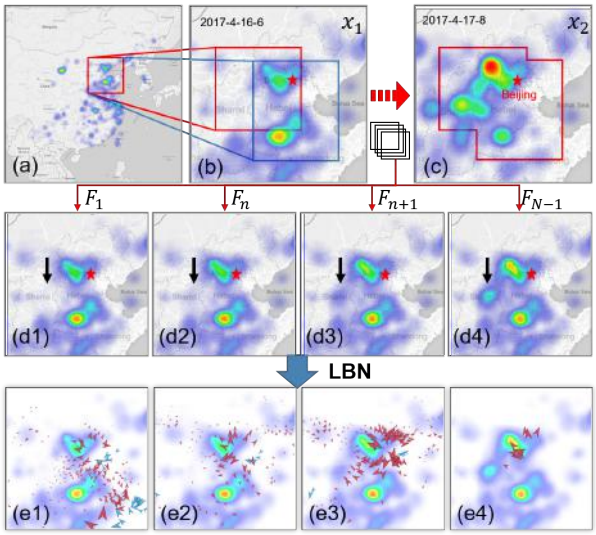


Fig. 10. The density map shows the air quality in the nation, and we focus on North China, which is influenced by haze. The 1st row is the real data around Beijing, the 2nd row shows the interpolation change between inputs, and the 3rd row presents change trend with arrows using LBN.

5.3 Air Quality Data

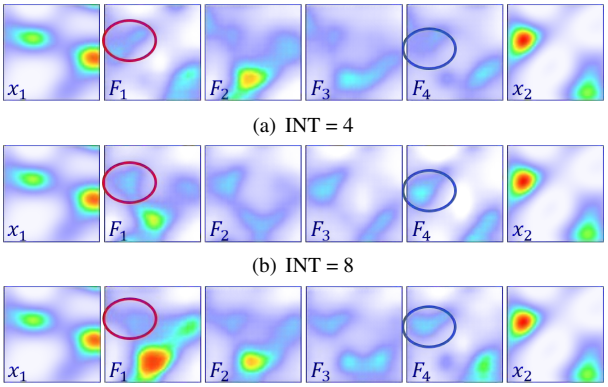
Big data visualization is often presented on a large scale, in these scenarios the large images are necessary. In this work, we collected the air quality index (AQI) in China, and the data are visualized by a heat map. Because air monitoring stations are not established everywhere, the information distributions are also dispersed geographically. In fact, users should only need to know the air change trend in parts of China, and it is easy to find that the most regions have no data. For showing information on a national scale and guaranteeing the generative quality, it is necessary to generate large-sized images. In this test, the full image is 1024×1024, which is commonly defined as a high-resolution image in the image processing domain. As we introduced in Section 4.3.2, our generative model uses 128×128 data for obtaining a good performance.

In recent years, it has been the focus of attention that haze has appeared frequently around Beijing in China [51]. Therefore, we take this region as the ROI, and Section 5.2 has determined the feasibility to infer the change process by GenerativeMap. However, the national information is too much to observe; therefore, Figure 10(a) is a typical visualization. Another characteristic is that the data are collected every two hours every day, which means that some regions would not change much compared with North China. We select the data from 06:00 on April 16, 2017, to 08:00 on April 17, 2017, and the AQI changes in the ROI. Figure 10(b) and Figure 10(c) present the records in the same ROI, and it is complex to perform visual analysis of the region independently with the traditional method. The GenerativeMap greatly reduces the difficulty of the task. As shown in Figure 10(d1-d4), the picture series blends the result of generated interpolations and the region map. The result is smooth and seamless, even when the data around the ROI show a slight change. The last generated frame F_{N-1} is similar to the target image as shown in Figure 10(c).

Considering that the data collected are not continuous, GenerativeMap can help us infer the probable process. Furthermore, the data change would be fuzzy with blending, which may make it difficult for users to identify the change. We can introduce LBN to show the differences in detail, as shown in Figure 10(e1-e4). A significant feature is that the red arrows occupy the main area, and the blue arrows decrease over time. The phenomena confirm the haze increases, as shown in Figure 10(b) and Figure 10(c). Another important feature is that the red arrows move from south to north, which presents the probable haze movement. LBN provides a clear and nonoverlapped distribution of the direct density change.

Table 2. Evaluation indexes

Index Symbol	Task
ICS	Measuring the similarity of continuous frames in a series.
TDS	Measuring the time cost of generating a specified number of interpolations.
INT	Describing how many interpolations we want to generate.
IGS	Evaluating how many features of input the interpolations contain. This comparison is made in an interpolation.
ILS	Evaluating whether our method can direct the overall transition trend correctly. We compare the generated frames and the ground truth to get this index.



(a) INT = 4

(b) INT = 8

(c) INT = 12

Fig. 11. A evaluation of interpolation ability. The visual effect is different when INT is different.

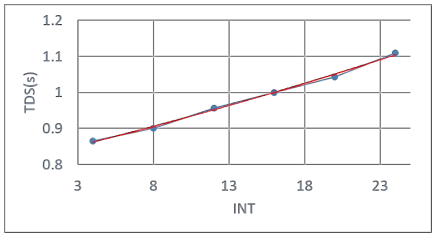


Fig. 12. The evaluation of INT-TDS. The TDS increases as the INT increases, the slope of curve is slight, and the time cost is available for generating the number of interpolations.

6 Evaluation and Discussion

GenerativeMap is tailored to allow users to solve tasks related to the extraction of density maps. As discussed in Section 1, the existing similar work was designed by scholars for their specialized tasks. Model-driven methods are available in these scenarios, which can be defined as problems of time segments and geometrical shape dimensions. Data-driven methods are designed for processing the coarse data or the data that has abnormal discretization. In contrast, our technology aims to help users to achieve references and to empower novice users to obtain density morphing by using existing data to train model, even when they have no experience in the domain.

For the deep learning model that needs pretraining, the pipeline uses the model as a generator, which generates interpolation samples according to stored parameters. For the trained model, the quick generation ability is the strength of deep learning. However, both quantitative and qualitative evaluation is still necessary. Combined with the characteristic of our pipeline, we design special evaluation methods, which help users to know the generative quality and efficiency. The evaluation indexes and tasks are shown in Table 2. The hash-based image similarity algorithm [40] is the basic measurement of in our evaluations, and all of the indexes are built on the algorithm. In our work, we use three hashes (*ahash*,*phash* and *dhash*) [24,49] because they can show good performances in different aspects.

6.1 Time Performance

In this work, we compare the generative quality by generating a different number of interpolation vectors, and we set an index that describes the number of targets (*INT*). The three series in Figure 11 represent *INT*=4, 8, and 12. Due to space limitations, we only present 4 interpolations that are selected from different *INT* series. The actual time cost of computing the interpolation is measured by another index, and we call it the time cost for different samples (*TDS*). The *TDS* index comparison is shown in Figure 12. The *TDS* would increase as the *INT* increases, and we can find that the slope of the curve is slight. In general, we do not need to generate too many interpolations, and the useful *INT* is often less than 24. The result confirms that the computational load does not explode when users need a more detailed process. A common issue in traditional methods for the extraction of more details is employing more parameters, while deep learning achieves this goal in the pretraining.

6.2 Generative Ability

The generative ability is evaluated from continuity and validity aspects. Figure 11(a) indicates a series of samples of *INT*=4, and the motion is not smooth and as continuous in the samples, particularly in the connection area (in the red circle). The disadvantage can also be seen in the blue circle. Considering the principles of GenerativeMap, the continuity of interpolations may be interrupted if the amount of samples is few, which seriously influences the visual effect. Another comparison is shown in Figure 11(b) and (c), where these samples are selected from *INT*=8 and *INT*=16, respectively. The final visual effect appears similar, and a further quantitative evaluation of interpolations is necessary.

We design evaluation indexes about the similarity computation of interpolated samples, which refer to the sample test in deep learning. We conduct experiments and design an index to compute continuity similarity (*ICS*) of one series. To measure the features our model learned, we separate the evaluation as a global one and a local one. The index of global similarity (*IGS*) describes the correlation of interpolation and input in one series, while the index of local similarity (ILS_i) is designed to describe the similarity of the *i*th frame and benchmark. The described method can be summarized as Equation 5:

$$\begin{cases} \Theta(P,Q) = \frac{1}{3}(ahashSim(P,Q) + phashSim(P,Q) + dhashSim(P,Q)) \\ ICS(F_n) = \frac{1}{INT-2}\sum_{i=2}^{INT}\Theta(F_i,F_{i-1}) \\ IGS(F_n) = \frac{1}{2(INT-2)}(\sum_{i=2}^{INT-1}\Theta(F_i,x1) + (\sum_{j=2}^{INT-2}\Theta(F_j,x2))) \\ ILS_i(F_n,G_n) = \frac{1}{j_2-j_1}\sum_{j_1=max(0,i-1)}^{j_2=min(i+1,INT)}\Theta(F_i,G_j) \end{cases} \tag{5}$$

where Θ is a mean similarity of two input images (P and Q). *ahashSim*, *phashSim* and *dhashSim* represent three similarity calculation methods based on three hash algorithms. F_n, G_n, $x1$, and $x2$ are all described in Table1.

All similarity indexes can be measured as scores, and higher scores mean that the generative performance is better. Figure 13 and Figure 15 show the indexes we used to compare details. As we expect, there is no linear correlation between the number and quality. The relationship of *IGS* and *INT* is shown in Figure 13, and the scores are not always increased as *INT* increases. We provide *ICS* in addition to *IGS*, and the growth of *ICS* slows down within more interpolations. *IGS* slightly decreases when *INT* is changed from 8 to 12. Hence, we conclude that feature loss appears if unnecessary number of interpolations are applied. Considering the performance and ability, *INT*=8 is an available choice for most of the scenarios.

It is difficult to select a proper scenario to validate ground truth. In our work, we mainly focus on the long-term discrete data and discon-

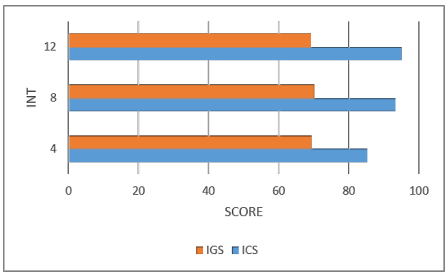
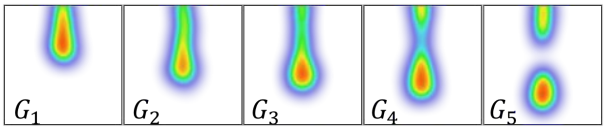
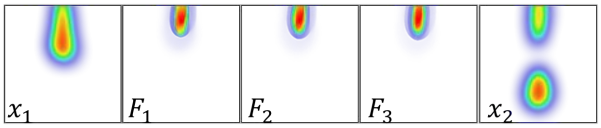


Fig. 13. The evaluation of IGS-ICS-INT. The IGS of INT=8 is the highest in all groups, and the growth of ICS slowdown within more INT, comprehensive conclude that INT=8 may be the best choice for most scenarios.

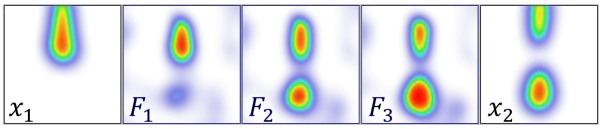
(a) The ground truth of droplet



(b) The density map of ground truth



(c) The morphing process with diffusion model



(d) The morphing process with GenerateMap

Fig. 14. The evaluation of realistic generative results by our pipeline. The highlighted area in (c) is sharp, which is obviously different from spheres. (d) is our generative result, which is more smooth and directs the change on the whole.

tinuous data records. An ideal ground truth should be a continuous phenomenon that can be directly observed. We choose the droplet movement as our experiment, which simulates the water tension. We compare the interpolations and water movement to evaluate the generative ability. The droplet movement contains a certain natural dynamic information. Figure 14(a) shows the dynamic process of the simulation, and we grab 5 frames from the process as the benchmarks, where the water droplet falls naturally [37]. The start-to-end frames of Figure 14(a) are the benchmark of our experiment. To evaluate the generative ability, we first create density maps, as shown in Figure14(b), by the density tool according to Figure 14(a). Figure 14(c) presents the result with the traditional diffusion model, and it is obvious that the model cannot solve the process correctly, which is shown in the middle 3 frames. By contrast, Figure 14(d) shows the probable motion of GenerativeMap, and it appears that the model infers the correct change gradually, particularly for the split process in F_3.

ILS means the difference of the generative interpolation and the real change process, and the index can help us to analyze the generative ability in addition to the visual effect. As shown in Figure 15, the scores will be higher if the generated result is more similar to the benchmark. An interesting result is that the generated *ILS* scores are more obvious than the diffusion *ILS* when the sample is close to the end benchmark, which follows the visual effect that we discussed above. The results show that the traditional diffusion model plays well at the start of motion. However, it is influenced by the data distribution and even fails.

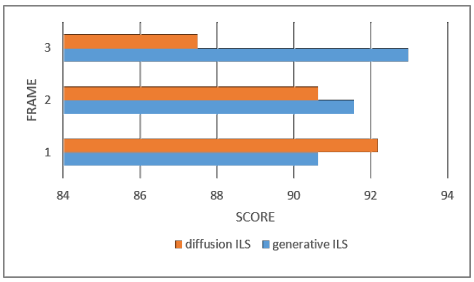


Fig. 15. The evaluation of the ILS-Frame. The generative ILS is higher than the diffusion ILS when the evaluated frames are close to the end frame.

The generated effect may be better if more information is provided or more interpolations are generated. The final evaluation results indicate that GenerativeMap has advantage in the integrated indicators.

6.3 Limitation

We evaluated our approach with a broad-based comparison that confirmed the effectiveness and necessity of our designs. We take PN to create training datasets; therefore, the model can learn natural movement features. However, the model has also learned the unnecessary noise. To avoid the noise, we have to introduce Gaussian blur into interpolations. Since we use many noise methods to normalize datasets; however, this means the method is insufficient if details of images are important.

Our system requires training the generative model, while most real data contain special physical or mathematical rules, such as in transportation and meteorology where their long-term activities follow regular patterns. These types of data need to be collected over a long continuous time. As a result, in the approach, we can only present limited random natural variation forms. For some fields, the movements follow some known rules, and the deep learning can be improved as presented in the work of Byungsoo et al. [20].

Another limitation is that our pipeline is still complex. A complex combination introduces an inconvenient implementation, which motivated us to optimize the pipeline in the future. As a well-known problem in deep learning, GANs are not models that are stable enough. The proposed pipeline can be further simplified and integrated to get an end-to-end model.

7 Conclusion and Future Work

We present a novel pipeline for extracting the dynamic density map that incorporates image processing methods and an improved generative model. The major contribution of our approach is to promote a general approach that helps users to obtain possible dynamics between selected density maps. In our work, we incorporate many models and design optimization for data evolution visualization, and in particular, we incorporate deep learning with regard to the information visualization.

An interesting avenue for future work is enhancing the visualization approach that utilizes the training data provided by our system. The research depends on the development of deep learning, combined with the conditions controlled by the generative model and the smooth generating process. The most anticipated expansion is that the approach might be incorporated within a spatiotemporal application that is usable by the general public.

Acknowledgments

The authors wish to acknowledge the support from NSFC under Grants (No. 61802128, 61672237, 61532002).

References

[1] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua. Cvae-gan: Fine-grained image generation through asymmetric training. In *2017 IEEE International Conference on Computer Vision*, pp. 2764–2773. IEEE, 2017.

[2] A. Biswas, G. Lin, X. Liu, and H.-W. Shen. Visualization of time-varying weather ensembles across multiple resolutions. *IEEE Transactions on Visualization and Computer Graphics*, 23:841–850, 2017.

[3] P. Bojanowski, A. Joulin, D. Lopez-Paz, and A. Szlam. Optimizing the latent space of generative networks. *arXiv*, abs/1707.05776, 2017.

[4] R. P. Botchen, D. Weiskopf, and T. Ertl. Texture-based visualization of uncertainty in flow fields. In *2005 IEEE Visualization*, pp. 647–654. IEEE, 2005.

[5] Z. I. Botev, J. F. Grotowski, D. P. Kroese, et al. Kernel density estimation via diffusion. *The annals of Statistics*, 38(5):2916–2957, 2010.

[6] J. Buchmuller, D. Jackle, E. Cakmak, U. Brandes, and D. A. Keim. Motionrugs: Visualizing collective trends in space and time. *IEEE Transactions on Visualization and Computer Graphics*, 25:76–86, 2018.

[7] J. J. Caban, A. Joshi, and P. Rheingans. Texture-based feature tracking for effective time-varying data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13:1472–1479, 2007.

[8] B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. Technical report, Lawrence Livermore National Lab., CA (United States), 1993.

[9] H. Chen, W. Chen, H. Mei, Z. Liu, K. Zhou, W. Chen, W. Gu, and K.-L. Ma. Visual abstraction and exploration of multi-class scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 20:1683–1692, 2014.

[10] W. Cui, X. Wang, S. Liu, N. H. Riche, T. M. Madhyastha, K. L. Ma, and B. Guo. Let it flow: a static method for exploring dynamic graphs. In *2014 IEEE Pacific Visualization Symposium*, pp. 121–128. IEEE, 2014.

[11] L. Dinh, D. Krueger, and Y. Bengio. Nice: Non-linear independent components estimation. *arXiv*, abs/1410.8516, 2014.

[12] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv*, abs/1605.09782, 2016.

[13] H. Doraiswamy, V. Natarajan, and R. S. Nanjundiah. An exploration framework to identify and track movement of cloud systems. *IEEE Transactions on Visualization and Computer Graphics*, 19:2896–2905, 2013.

[14] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. C. Courville. Adversarially learned inference. *arXiv*, abs/1606.00704, 2016.

[15] D. Fortun, P. Bouthemy, and C. Kervrann. Optical flow modeling and computation: A survey. *Computer Vision and Image Understanding*, 134:1–21, 2015.

[16] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial networks. *arXiv*, abs/1406.2661, 2014.

[17] H. Guo, F. Hong, Q. Shu, J. Zhang, J. Huang, and X. Yuan. Scalable lagrangian-based attribute space projection for multivariate unsteady flow data. In *2014 IEEE Pacific Visualization Symposium*, pp. 33–40. IEEE, 2014.

[18] C. Heine, H. Leitte, M. Hlawitschka, F. Iuricich, L. De Floriani, G. Scheuermann, H. Hagen, and C. Garth. A survey of topology-based methods in visualization. *Computer Graphics Forum*, 35(3):643–667, 2016.

[19] H. Hoppe. Smooth view-dependent level-of-detail control and its application to terrain rendering. In *1998 Proceedings Visualization*, pp. 35–42. IEEE, 1998.

[20] B. Kim and D. Aguilar-Cázares. Robust reference frame extraction from unsteady 2d vector fields with convolutional neural networks. *arXiv*, abs/1903.10255, 2019.

[21] S. Kim, S. Jeong, I. Woo, Y. Jang, R. Maciejewski, and D. Ebert. Data flow analysis and visualization for spatiotemporal statistical data without trajectory information. *IEEE Transactions on Visualization and Computer Graphics*, 24:1287–1300, 2017.

[22] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv*, abs/1312.6114, 2013.

[23] V. Krasnopolsky, S. Nadiga, A. Mehra, E. Bayler, and D. Behringer. Neural networks technique for filling gaps in satellite measurements: Application to ocean color observations. *Computational Intelligence and Neuroscience*, 2016:1–9, 01 2016.

[24] N. Krawetz. Kind of like that. `http://www.hackerfactor.com/blog/index.php?/archives/529-Kind-of-Like-That.html`. Accessed Jan 21, 2013.

[25] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv*, abs/1512.09300, 2015.

[26] H. Li, L.-Y. Wei, P. V. Sander, and C.-W. Fu. Anisotropic blue noise sampling. *ACM Transactions on Graphics*, 29(6):167, 2010.

[27] L. Liu, L. M. K. Padilla, S. H. Creem-Regehr, and D. H. House. Visualizing uncertain tropical cyclone predictions using representative samples from ensembles of forecast tracks. *IEEE Transactions on Visualization and Computer Graphics*, 25:882–891, 2018.

[28] L. Liu, D. Silver, K. G. Bemis, D. Kang, and E. Curchitser. Illustrative visualization of mesoscale ocean eddies. *Computer Graphics Forum*, 36:447–458, 2017.

[29] L. Y. Liu, A. P. Boone, I. T. Ruginski, L. M. K. Padilla, M. Hegarty, S. H. Creem-Regehr, W. B. Thompson, C. Yuksel, and D. H. House. Uncertainty visualization by representative sampling from prediction ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 23:2165–2178, 2017.

[30] Z. Peng and R. S. Laramee. Higher dimensional vector field visualization: A survey. In *2009 Theory and Practice of Computer Graphics*, pp. 149–163. The Eurographics Association, 2009.

[31] K. Perlin. Improving noise. *ACM Transactions on Graphics*, 21(3):681–682, 2002.

[32] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv*, abs/1511.06434, 2016.

[33] M. Rautenhaus, S. Zafirovski, S. Siemen, R. Hoffman, R. M. Kirby, M. Mirzargar, M. Durande, and R. Westermann. Visualization in meteorologya survey of techniques and tools for data analysis tasks. *IEEE Transactions on Visualization and Computer Graphics*, 24:3268–3296, 2018.

[34] H. Ren, E. Cromwell, B. Kravitz, and X. Chen. Using deep learning to fill spatio-temporal data gaps in hydrological monitoring networks. *Hydrology and Earth System Sciences Discussions*, pp. 1–20, 05 2019.

[35] V. A. Sindagi and V. M. Patel. Generating high-quality crowd density maps using contextual pyramid cnns. In *2017 IEEE International Conference on Computer Vision*, pp. 1879–1888. IEEE, 2017.

[36] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. In *2016 Proceedings of the Thirty AAAI Conference on Artificial Intelligence*, pp. 4278–4284. AAAI Press, 2016.

[37] N. Thürey, C. Wojtan, M. Gross, and G. Turk. A multiscale approach to mesh-based surface tension flows. *ACM Transactions on Graphics*, 29(4):48, 2010.

[38] H.-F. Tsai, J. Gajda, T. F. Sloan, A. Rares, and A. Q. Shen. Usiigaci: Instance-aware cell tracking in stain-free phase contrast microscopy enabled by machine learning. *SoftwareX*, 9:230–237, 2019.

[39] A. A. Valsangkar, J. M. Monteiro, V. Narayanan, I. Hotz, and V. Natarajan. An exploratory framework for cyclone identification and tracking. *IEEE Transactions on Visualization and Computer Graphics*, 25:1460–1473, 2018.

[40] R. Venkatesan, S.-M. Koon, M. H. Jakubowski, and P. Moulin. Robust image hashing. In *2000 Proceedings of The International Conference on Image Processing*, vol. 3, pp. 664–666. IEEE, 2000.

[41] T. Von Landesberger, F. Brodkorb, P. Roskosch, N. Andrienko, G. Andrienko, and A. Kerren. Mobilitygraphs: Visual analysis of mass mobility dynamics via spatio-temporal graphs and clustering. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):11–20, 2016.

[42] J. Wang, S. Hazarika, C. Li, and H.-W. Shen. Visualization and visual analysis of ensemble data: A survey. *IEEE Transactions on Visualization and Computer Graphics*, PP(99), 2018.

[43] L.-Y. Wei. Multi-class blue noise sampling. *ACM Transactions on Graphics*, 29(4):79, 2010.

[44] J. Woodring, J. P. Ahrens, J. Figg, J. Wendelberger, S. Habib, and K. Heitmann. In-situ sampling of a large-scale particle simulation for interactive visualization and analysis. *Computer Graphics Forum*, 30:1151–1160, 2011.

[45] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *2016 Advances in neural information processing systems*, pp. 82–90, 2016.

[46] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu. Opinionflow: Visual analysis of opinion diffusion on social media. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1763–1772, 2014.

[47] D.-M. Yan, J. Guo, B. Wang, X. Zhang, and P. Wonka. A survey of blue-noise sampling and its applications. *Journal of Computer Science and Technology*, 30:439–452, 2015.

[48] H. Yu, C. Wang, C.-K. Shene, and J. Chen. Hierarchical streamline

bundles. *IEEE Transactions on Visualization and Computer Graphics*, 18:1353–1367, 2012.

[49] C. Zauner. Implementation and benchmarking of perceptual image hash functions. Master's thesis, Upper Austria University of Applied Sciences, Hagenberg Campus, 2010.

[50] J. Zhang, Y. Zheng, and D. Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *2017 Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 1655–1661. AAAI Press, 2017.

[51] W. Zhang, Y. Wang, Q. Zeng, Y. Wang, G. Chen, T. Niu, C. Tu, and Y. Chen. Visual analysis of haze evolution and correlation in beijing. *Journal of Visualization*, 22:161–176, 2019.